UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ENVPREDRSCHFAC Technical Paper No. 5-74	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Three-Parameter Model for Limited Area Forecasting		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Dr. L. Bengtsson Swedish Meteorological and Hydrological Institute, Stockholm, Sweden		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Environmental Prediction Research Facility Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Commander, Naval Air Systems Command Department of the Navy Washington, D.C. 20361		12. REPORT DATE March 1974
		13. NUMBER OF PAGES 112
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; Distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Atmosphere
Numerical
Atmospheric Circulation
Atmospheric Prediction
Limited Area Forecasting

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes an operational quasi-geostrophic three-parameter model. The original model was developed by Dr. L. Bengtsson and has been used operationally for several years at the Swedish Meteorological and Hydrological Institute.

The improved model described in this report incorporates an Ekman function and the effect of the flow over mountains as well as sensible and latent heat sources. Humidity and precipitation

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. (continued)

are also predicted by the model.

An additional feature is the optional inclusion in the vorticity equation of the terms which are usually considered negligible: (1) the advection of vorticity by the divergent wind, (2) the product of relative vorticity and divergence, (3) the vertical advection of vorticity, and (4) the twisting term.

The model can easily be adapted for different geographical areas with different grid lengths and also be integrated over different vertical layers. The lateral boundary values can be fixed or allowed to vary in time and thus it is possible to apply the so called nesting or telescoping technique. A special, relatively simple method to apply the nesting techniques is described.

AN (1) AD- 777 406
FG (2) 040200
FG (2) 120500
CI (3) (U)
CA (5) ENVIRONMENTAL PREDICTION RESEARCH FACILITY (NAVY)
MONTEREY CALIF
TI (6) A Three-Parameter Model for Limited Area Forecasting,
TC (8) (U)
AU (10) Bengtsson,L.
RD (11) Mar 1974
PG (12) 113p
RS (14) ENVPREDRSHCHFAC-tech-paper-5-74
RC (20) Unclassified report
DE (23) *Weather forecasting, *Computer programs, FORTRAN,
Numerical analysis, Atmospheric motion, Vortices,
Humidity
DC (24) (U)
ID (25) *Numerical weather forecasting, FORTRAN 4 programming
language, PROG 3P computer program, STEP 3P computer
program, STEPEXT computer program, Primitive equations,
Sensible heat, Latent heat
IC (26) (U)
AB (27) The report describes an operational quasi-geostrophic
three-parameter model. The original model was developed
by Dr. L. Bengtsson and has been used operationally for
several years at the Swedish Meteorological and
Hydrological Institute. The improved model described in
this report incorporates an Ekman function and the
effect of the flow over mountains as well as sensible
and latent heat sources. Humidity and precipitation are
also predicted by the model. (Modified author abstract)
AC (28) (U)
DL (33) 01
CC (35) 407279

ENVIRONMENTAL PREDICTION RESEARCH FACILITY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93940

IN REPLY REFER TO
EPRF/KGR:wc
5605
Ser: 105
4 APR 1974

From: Commanding Officer
To: Distribution List

Subj: Research Publication; forwarding of

Encl: (1) "A Three-Parameter Model For Limited Area Forecasting," ENVPREDRSCHFAC Technical Paper No. 5-74, March 1974

1. Enclosure (1) is forwarded for information.

G. D. HAMILTON

Distribution List:
Environmental Prediction Research Facility Master Distribution List of 1 September 1973.

LIST II: SNDL Nos. A2A (Codes 410, 412, 480, 482, 460M and 462 only), A3 (OP-986G and OP-945 only), A4A (MAT-03416 only), B2 (RADM Kotsch only), E3A (Code 2620 only), E3B, FD1 (less Code N-5), FF38 (ENVSCIDEPT only), FF42 (less Oceanography Dept.), FKA1A (AIR-5017, AIR-370C and AIR-05F only), FW1 (Codes 304 and 37 only), FW2 (CO FLENUMWEACEN only), FW3 and FW4 (Suitland and London only).

LIST III: Item Nos. 1, 2, 3, 5, 6 and 7.

LIST IV: Item Nos. 3 and 7.

LIST V: Item Nos. 1 and 2.

LIST VII.a: Item Nos. 1, 3, 5, 8, 14 and 15.
VII.d: Item No. 5.

LIST IX.a: Item Nos. 1, 2, 3, 10, 17, 22 and 32.
IX.b: Item No. 2.

LIST X: Item Nos. 4, 8, 14 and 32.

LIST XII: Item Nos. 1, 2, 5, 6 and 7.

LIST XIII: Item Nos. AUS-1, 5 (including Meteorology Research Center), CAN-3, DEN-1, ENG-1,5 and 14, FIN-1, FRA-1, GER-1 and 3, HGKG-1, INDIA-4 (including Meteor. Dept., New Delhi), ITL-2, JAP-1, NZEA-1, NOR-1 and 3, SOAF-1 and 2, SWE-1, 2, 3 and 4.

ENVPREDRSCHFAC
Technical Paper No. 5-74

A THREE-PARAMETER MODEL FOR LIMITED AREA FORECASTING

by

DR. L. BENGTSSON

MARCH 1974

ENVIRONMENTAL PREDICTION RESEARCH FACILITY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93940

CONTENTS

ACKNOWLEDGEMENTS 2

1. INTRODUCTION 3

2. PROGNOSTIC EQUATIONS 7

3. BOUNDARY CONDITION 15
   3.1 Lower Boundary Condition 15
   3.2 Lateral Boundary Condition 16

4. PARAMETERIZATION OF PHYSICAL PROCESSES 18
   4.1 Sensible Heat 18
   4.2 Prognostic Equation for Humidity 19
   4.3 Latent Heat and Computation of Precipitation 21

5. COMPUTATION OF VERTICAL MOTION 22

6. NUMERICAL VALUES OF DIFFERENT CONSTANTS 24

7. INTEGRATION OF THE COMPLETE VORTICITY EQUATION 25
   7.1 General Aspects 25
   7.2 Derivation of the Forecasting Equations 29

8. NUMERICAL SOLUTION OF THE 3-PARAMETER MODEL 33
   8.1 General Aspects 33
   8.2 Finite Differences 34
   8.3 Computation of Sensible Heat 36
   8.4 Computation of Latent Heat 37
   8.5 Numerical Solution of the Non-geostrophic Terms 40

9. INITIALIZATION 48
   9.1 Numerical Solution 49
   9.2 Initialization of the Specific Humidity 50

10. TIME INTEGRATION AND TREATMENT OF BOUNDARY CONDITIONS 51

11. THE CRITERION OF ELLIPTICITY 54

REFERENCES 55

APPENDIX A: GENERAL PRINCIPLES OF PROGRAMMING 58
APPENDIX B: PROGRAM SPECIFICATIONS 61
APPENDIX C: PROGRAM LISTINGS 92

ACKNOWLEDGEMENTS

The author wishes to thank LT Byron Maxwell, USN for his careful computations and critical checking of the manuscript. The efforts of Mrs. Winona Carlisle who typed the manuscript are also gratefully acknowledged.

I. INTRODUCTION

Numerical weather prediction with the aid of primitive equations has now been performed operationally for several years (Reiser, 1969; Shuman, 1968). Forecasts for one and two days by primitive equations indicate some improvement when compared with forecasts with the quasi-geostrophic and filtered equations. However, it has not been shown in a clear and convincing way whether this is solely due to the unfiltered part of the equations or to purely numerical improvements such as improved vertical resolution, an alternating grid, and the introduction of new physical effects such as sensible heat, latent heat, radiation, etc. When the forecasts with the primitive models are extended further in time, the improvements achieved with these models seem to be more obvious.

If a quasi-geostrophic model is compared with a primitive model with a resolution of three to four vertical layers or less, it will be found that the computational time (and cost) for the forecast with the primitive model is roughly about ten times as large as for the quasi-geostrophic model, if explicit time-integration schemes are used. The grid lengths now in use for operational weather prediction are still about 300 to 400 km.

Since significant weather disturbances have dimensions which are only three to five times that size, it is obvious that the truncation errors in the computation of the horizontal finite differences are much too large.

That grid distances of 300 km and more have been used for such a long time is naturally due to insufficient computational capacity, but may also partly be due to accustomed routine. However, a decrease in the horizontal grid length to half the size implies an increase in the number of grid points by a factor of four for the same computation area.

Due to the criterion of Courant, Friedrichs and Lewy, the time step must be decreased by a factor of two in order to maintain computational stability. If the computations can be organized in the same way as for the larger grid, a halving of the grid length, therefore, implies an eight-fold increase in the computation time. From the operational point of view, therefore, a primitive model should be compared with a quasi-geostrophic model where the horizontal grid length has been decreased to half the size.

When the forecasting areas are small and cover only one third or less of a hemisphere, the horizontal boundaries will fall in meteorologically active areas. This disadvantage does not create any large problem for the filtered models and a moderate horizontal smoothing in the neighborhood of the boundaries is sufficient. For the primitive models the

problem is much worse, since high-amplitude gravity waves are generated at the boundaries. These waves propagate into the area and greatly affect the meteorological information. Except for some successful experiments (Bushby, 1967, 1968; Gerrity, 1969), there has not been reported any adequate technique to avoid this in the general case. For this reason, forecasts for restricted areas with the complete equations in operational use imply considerable difficulties.

It may now be argued that it is of no use to apply a quasi-geostrophic model to a fine mesh, since that means the model will predict (or try to predict) scales of motion characterized by large Rossby numbers. For instance, when the Rossby number is on the order of one, all the terms in the vorticity equation are of equal magnitude. The quasi-geostrophic models will thus, according to this analysis, give rise to intolerably large errors for small and intense vortices, especially at low latitudes. However, it has been shown (Bengtsson and Moen, 1971) that substantial improvements in forecasts from quasi-geostrophic models are obtained if the grid size is reduced from 300 to 150 km.

The reason for this is that the higher order terms in the vorticity equation to a considerable degree cancel each other. It is only in the final stage of the cyclone development, when the flow becomes very deformed, that these terms become important.

It is also possible, as will be shown in this report, to include these terms in the integration and estimate them with the aid of quasi-geostrophic divergence.

It is the experience of the author that filtered models still are very useful for short-range predictions at medium and high latitudes. It is also very probable that unsuccessful predictions of especially rapid cyclogenesis are due to inaccuracies in the initial state and to unsatisfactory ways of including topographical effects, parameterization of dissipation, and the heating mechanisms. Recent comparisons performed in Sweden between filtered and primitive equation models support this view.

2. PROGNOSTIC EQUATIONS

The vorticity equation, thermodynamical equation and the continuity equation read:

$$\frac{\partial \zeta}{\partial t} = - \mathbb{V} \cdot \nabla \eta - f \quad D; \tag{2.1a}$$

$$\frac{\partial}{\partial t} \left(\frac{\partial \phi}{\partial p}\right) = - \mathbb{V} \cdot \nabla \left(\frac{\partial \phi}{\partial p}\right) - R(\Gamma_d - \Gamma) \frac{\omega}{p} - \frac{R}{c_p p} \left(\frac{\delta Q}{dt}\right); \tag{2.1b}$$

$$\frac{\partial \omega}{\partial p} = - D; \tag{2.1c}$$

where

$$\mathbb{V} = \mathbb{k} X \nabla \phi,$$

$$D = \nabla \cdot \mathbb{V},$$

$$\Gamma_d = \frac{1}{\rho c_p} \text{, and}$$

$$\Gamma = \frac{\partial T}{\partial p} .$$

We will now introduce some special model assumptions. We assume that the atmosphere is bounded by a pressure surface near the surface of the earth, $p=p_o$, and an upper pressure surface, $p=p_1$, near the tropopause. We will further assume a third interjacent pressure surface, $p=p_m$, which separates the atmosphere in two layers. We will now represent the wind field in the following way:

$$\mathbb{V} = \mathbb{V}_m - 2\mathbb{V}_1 \frac{p-p_m}{p_o-p_m} \quad ; \quad \text{layer 1}$$

$$\mathbb{V} = \mathbb{V}_m + 2\mathbb{V}_2 \frac{p_m-p}{p_m-p_1} \quad ; \quad \text{layer 2}$$

$$\mathbb{V} = (\mathbb{V}_m + 2\mathbb{V}_2) \frac{p}{p_1} \quad ; \quad \text{layer 3}$$

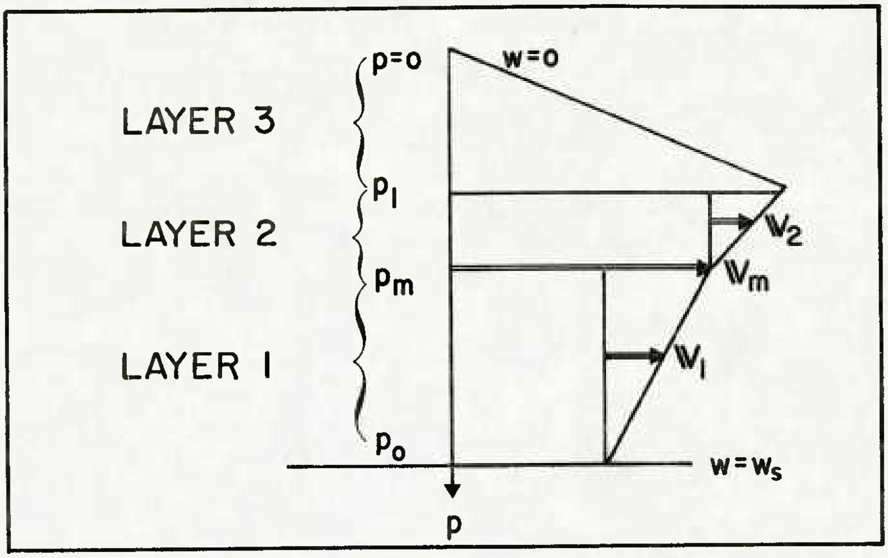


Figure 1. Vertical representation of the wind for a 3-parameter model.

According to the definition, equations for D and ζ will have the same form as those for $\mathbb{V}$. Equation (2.1a) can now be integrated between $p=p_o$ and $p=0$ with the boundary conditions $\omega(p=0)=0$ and $\omega(p_o)=\omega_s$.

This gives a prognostic equation for the vertically integrated vorticity:

$$\frac{\partial}{\partial t}(\zeta_m + c_1\zeta_2 - c_2\zeta_1) = - \mathbb{V}_m\cdot\nabla(c_3\eta_m - c_2\zeta_1 + c_4\zeta_2 + c_7 f)$$

$$- \mathbb{V}_1\cdot\nabla(-c_2\eta_m + c_5\zeta_1) - \mathbb{V}_2\cdot\nabla(c_4\eta_m + c_6\zeta_2 + 2c_7 f) + c_8 f\omega_s \qquad (2.2)$$

where

$$c_1 = \frac{2p_m}{2p_o-p_1} \qquad c_4 = \frac{6p_m-2p_1}{6p_o-3p_1} \qquad c_7 = \frac{p_1}{6p_o-3p_1}$$

$$c_2 = \frac{2(p_o-p_m)}{2p_o-p_1} \qquad c_5 = \frac{8p_o-8p_m}{6p_o-3p_1} \qquad c_8 = \frac{2}{2p_o-p_1}$$

$$c_3 = \frac{6p_o-4p_1}{6p_o-3p_1} \qquad c_6 = \frac{8p_m}{6p_o-3p_1} .$$

The next two prognostic equations are computed from the difference between the vorticity equation at level p_m and p_o and the corresponding difference for p_1 and p_m. We will now get two vorticity equations valid for layer 1 and layer 2:

$$\frac{\partial\zeta_1}{\partial t} + (\mathbb{V}_m-\mathbb{V}_1)\cdot\nabla\zeta_1 + \mathbb{V}_1\cdot\nabla(\eta_m-\zeta_1) + f\ D_1 = 0; \qquad (2.3)$$

$$\frac{\partial\zeta_2}{\partial t} + (\mathbb{V}_m+V_2)\cdot\nabla\zeta_2 + \mathbb{V}_2\cdot\nabla(\eta_m+\zeta_2) + f\ D_2 = 0. \qquad (2.4)$$

From these two equations we will now eliminate the divergencies D_1 and D_2 with the aid of the continuity equation (2.1c) and the thermodynamical equation (2.1b). We will first integrate the continuity equation between the two levels p_a and p_b:

$$\omega(p_a) = \omega(p_b) + \int_{p_a}^{p_b} D\ dp. \qquad (2.5)$$

We will now put $p_a = p$, $p_b = p_o$ and $\omega(p_o) = \omega_s$ and will, thereby, get an expression for $\omega(p)$ in layer 1:

$$\omega(p) = \omega_s + (p_o - p)\, D_m + \frac{(p-p_m)^2 - (p_o-p_m)^2}{p_o - p_m}\, D_1. \qquad (2.6)$$

With $p_a = p$, $p_b = p_m$ and $\omega(p_m)$ from (2.6) we get the following expression for ω in layer 2:

$$\omega(p) = \omega_s + (p_o - p)\, D_m - (p_o - p_m)\, D_1 + \frac{(p_m - p)^2}{p_m - p_1}\, D_2. \qquad (2.7)$$

With $p_a = p$, $p_b = 0$ and $\omega(p=0) = 0$ we will have for layer 3:

$$\omega(p) = -\frac{p^2}{2p_1}\,(D_m + 2D_2). \qquad (2.8)$$

Finally we get a relation between D_m, D_1, D_2 and ω_s with the aid of an integration of the continuity equation from the top to the bottom of the atmosphere:

$$D_m = c_2 D_1 - c_1 D_2 - c_8 \omega_s. \qquad (2.9)$$

We now introduce the stream function into the thermodynamical equation and integrate through layers 1 and 2. $(\Gamma_d - \Gamma)$ is assumed to be constant in every layer and $\left(\frac{\delta Q}{dt}\right)_{pm} = \left(\frac{\delta Q}{dt}\right)_{p_1} = 0$.

$$2f \frac{\partial \psi_1}{\partial t} = -2f\mathbb{V}_m \cdot \nabla \psi_1 + R(\Gamma_d - \Gamma)_1 \int_{p_m}^{p_o} \frac{\omega}{p}\, dp + \frac{R}{2c_p} \ln\left(\frac{p_o}{p_m}\right)\left(\frac{\delta Q}{dt}\right)_{p_o} ;$$

$$2f \frac{\partial \psi_2}{\partial t} = -2f\mathbb{V}_m \cdot \nabla \psi_2 + R(\Gamma_d - \Gamma)_2 \int_{p_1}^{p_m} \frac{\omega}{p}\, dp . \tag{2.10}$$

The integrals $\int \frac{\omega}{p}\, dp$ can be computed with the aid of equations (2.6), 2.7) and (2.9), and the system (2.10) can be written

$$\begin{aligned} m_1 D_1 + m_2 D_2 &= H_1 ; \\ n_1 D_1 + n_2 D_2 &= H_2 ; \end{aligned} \tag{2.11}$$

where

$$m_1 = \frac{R}{2} (\Gamma_d - \Gamma)_1 \left[\frac{p_1 (p_o - p_m)^2 + p_m^2 (2p_o - p_1)}{(p_o - p_m)(2p_o - p_1)} \cdot \ln\left(\frac{p_o}{p_m}\right) \right.$$

$$\left. - \frac{2(p_o - p_m)^2}{2p_o - p_1} + \frac{1}{2}(p_o + p_m) - 2p_m \right] ;$$

$$m_2 = \frac{R}{2} (\Gamma_d - \Gamma)_1 \left[- \frac{2p_o p_m}{2p_o - p_1} \cdot \ln\left(\frac{p_o}{p_m}\right) + \frac{2p_m (p_o - p_m)}{2p_o - p_1} \right] ;$$

$$n_1 = \frac{R}{2} (\Gamma_d - \Gamma)_2 \left[\frac{p_1 (p_o - p_m)}{2p_o - p_1} \ln\left(\frac{p_m}{p_1}\right) - \frac{2(p_o - p_m)(p_m - p_1)}{2p_o - p_1} \right] ;$$

$$n_2 = \frac{R}{2} (\Gamma_d - \Gamma)_2 \left[\frac{p_1 p_m (2p_o - p_m)}{(2p_o - p_1)(p_m - p_1)} \ln\left(\frac{p_m}{p_1}\right) + \frac{2p_m (p_m - p_1)}{2p_o - p_1} \right.$$

$$\left. + \frac{1}{2}(p_m + p_1) - 2p_m \right];$$

$$H_1 = f \frac{\partial \psi_1}{\partial t} + f\mathbb{V}_m \cdot \nabla \psi_1 - \frac{R}{4c_p} \ln\left(\frac{p_o}{p_m}\right) \cdot \left(\frac{\delta Q}{dt}\right)_{p_o}$$

$$- \frac{R}{2} (\Gamma_d - \Gamma)_1 \left[- \frac{p_1}{2p_o - p_1} \ln\left(\frac{p_o}{p_m}\right) + \frac{2(p_o - p_m)}{2p_o - p_1} \right] \omega_s \;;$$

$$H_2 = f \frac{\partial \psi_2}{\partial t} + f\mathbb{V}_m \cdot \nabla \psi_2 - \frac{R}{2} (\Gamma_d - \Gamma)_2 \left[- \frac{p_1}{2p_o - p_1} \ln\left(\frac{p_m}{p_1}\right) \right.$$

$$\left. + \frac{2(p_m - p_1)}{2p_o - p_1} \right] \omega_s .$$

From the system (2.11) we can easily express the divergencies in an explicit way:

$$D_1 = -a_1 H_1 + a_2 H_2;$$
$$D_2 = b_1 H_1 - b_2 H_2; \qquad (2.12)$$

where

$$a_1 = \frac{-n_2}{m_1 n_2 - n_1 m_2}; \qquad a_2 = \frac{-m_2}{m_1 n_2 - n_1 m_2};$$

$$b_1 = \frac{-n_1}{m_1 n_2 - n_1 m_2}; \qquad b_2 = \frac{-m_1}{m_1 n_2 - n_1 m_2}.$$

Introducing these expressions for D_1 and D_2 into the prognostic equations (2.3) and (2.4) and expressing the wind and vorticity in terms of the stream function gives:

$$\nabla^2 \frac{\partial\psi_1}{\partial t} - a_1 f^2 \frac{\partial\psi_1}{\partial t} + a_2 f^2 \frac{\partial\psi_2}{\partial t} = - J(\psi_m;\zeta_1) - J(\psi_1;n_m-2\zeta_1)$$

$$+ a_1 f^2 J(\psi_m;\psi_1) - a_2 f^2 J(\psi_m;\psi_2) - a_3 f\omega_S - a_1 fH \qquad (2.13)$$

$$\nabla^2 \frac{\partial\psi_2}{\partial t} - b_2 f^2 \frac{\partial\psi_2}{\partial t} + b_1 f^2 \frac{\partial\psi_1}{\partial t} = - J(\psi_m;\zeta_2) - J(\psi_2;n_m+2\zeta_2)$$

$$- b_1 f^2 J(\psi_m;\psi_1) + b_2 f^2 J(\psi_m;\psi_2) + b_3 f\omega_s + b_1 fH \qquad (2.14)$$

Here we have

$$a_3 = a_1 s_1 - a_2 s_2;$$

$$b_3 = b_1 s_1 - b_2 s_2;$$

where

$$s_1 = \frac{R}{2}(\Gamma_d-\Gamma)_1 \left[- \frac{p_1}{2p_o-p_1} \ln(\frac{p_o}{p_m}) + \frac{2(p_o-p_m)}{2p_o-p_1} \right];$$

$$s_2 = \frac{R}{2}(\Gamma_d-\Gamma)_2 \left[- \frac{p_1}{2p_o-p_1} \ln(\frac{p_m}{p_1}) + \frac{2(p_m-p_1)}{2p_o-p_1} \right];$$

$$H = \frac{R}{4c_p} \ln(\frac{p_o}{p_m}) (\frac{\delta Q}{dt})_{p_o} = h_1 (\frac{\delta Q}{dt})_{p_o} \text{ with } h_1 = \frac{R}{4c_p} \ln(\frac{p_o}{p_m}).$$

We will also introduce the stream functions into equation (2.2) and define ψ_M by $\psi_M=\psi_m - c_2\psi_1+c_1\psi_2$.

This results in the equation

$$\nabla^2 \frac{\partial \psi_M}{\partial t} = - J(\psi_m; c_3 n_m - c_2 \zeta_1 + c_4 \zeta_2 + c_7 f) - J(\psi_1; -c_2 n_m + c_5 \zeta_1)$$

$$- J(\psi_2; c_4 n_m + c_6 \zeta_2 + 2c_7 f) + c_8 f \omega_s . \tag{2.15}$$

The equations (2.13) through (2.15) now constitute our system of prognostic equations. The only thing which we now have to do is to find an expression for the non-adiabatic heat H, and the lower boundary condition, ω_s.

3. BOUNDARY CONDITIONS

3.1 LOWER BOUNDARY CONDITIONS

We now assume that ω_s can be separated into two parts; one part which depends upon the dissipation in the boundary layer ω_s' and one part which depends upon topography ω_s''. We thereby assume:

$$\omega_s = \omega_s' + \omega_s'' \; . \tag{3.1.1}$$

From the Ekman theory about the variation of the wind in the friction layer, we could easily derive the following expression:

$$\omega_s' = - g\rho_o \sqrt{\frac{K}{2f}} \cdot F \cdot \zeta_o \quad \text{where } F = 1 + c\cdot\sin\Theta - c\cdot\cos\Theta,$$

the wind in the surface layer has a magnitude $c|\mathbb{V}_o|$, the angle between the geostrophic wind and the surface wind is given by Θ, K is the turbulent coefficient of the viscosity, and ρ_o is the density of the air at the surface. Inserting the following numerical values:

$K = 10$ m/s; $g = 9.81$ m/s^2; $f = 10^{-4}s^{-1}$; $T_o = 280°K$;

$R = 287$; $p_o = 100$ cb; gives

$$\omega_s' = - 2.729597\, F\cdot\zeta_o = -2.729597\cdot F(\zeta_m - 2\zeta_1) . \tag{3.1.2}$$

$F(c,\Theta)$ can be given a constant value in the model or we can also assume different values over land and sea.

The following values will be used:

Over land $\Theta = 10°$ and $c = 0.78$ gives $F = 0.36635$,

that is $\omega_s' = -1.\zeta_o$.

Over sea $\Theta = 5°$; $c = 0.85$ gives $F = 0.22734$,

that is, $\omega_s' = 0.62055\ \zeta_o$.

If k_2 is assumed to be that part of the air which is forced over the mountains, we get

$$\omega_s'' = k_2\ \mathbb{V}_o \cdot \nabla p_s = k_2\ (\mathbb{V}_m - 2\mathbb{V}_1) \cdot \nabla p_s \qquad (3.1.3)$$

$k_2 = 1$ will be used in the model.

The equations (3.1.1 - 3.1.3) now give the lower boundary condition for ω_s. For further information see Bengtsson (1969). This way of treating the topographical effect as given by equation 3.1.3 is quite unrealistic and seems to underestimate the effect of the topography. (This will be especially true for steep and/or small scale mountains.) A new way to treat mountains as impenetrable vertical barriers has recently been published (Egger, 1972). A similar way to include mountains in vertically integrated balanced models will be described by the author in a coming investigation.

3.2 LATERAL BOUNDARY CONDITIONS

The model can use two different kinds of lateral boundary conditions namely:

a. <u>Constant Inflow</u>

$$\frac{\partial \psi_1}{\partial t} = \frac{\partial \psi_2}{\partial t} = \frac{\partial \psi_m}{\partial t} = 0$$

$$\begin{aligned} \zeta_1(t) &= \zeta_1(t=0) \quad (= \text{constant in time}) \\ \zeta_2(t) &= \zeta_2(t=0) \quad (= \text{constant in time}) \\ \zeta_m(t) &= \zeta_m(t=0) \quad (= \text{constant in time}) \end{aligned} \tag{3.2.1}$$

b. <u>Variable Inflow</u>

The values for $\frac{\partial \psi_1}{\partial t}$, $\frac{\partial \psi_2}{\partial t}$, $\frac{\partial \psi_m}{\partial t}$, ζ_1, ζ_2, and ζ_m along the boundary are generated through an integration for a larger area which includes the actual area. Interpolation in time and space is necessary to syncronize the boundary values. A technical description of this is found in section 10.

4. PARAMETERIZATION OF PHYSICAL PROCESSES

4.1 SENSIBLE HEAT

In the computation of equation (2.10) we assume that $\frac{\delta Q}{dt}$ decreased linearly to 0 from p_o to p_m. The sensible heat which is transported to the atmosphere from the underlying surface, is introduced in the following way:

$$\left(\frac{\delta Q}{dt}\right)_{p_o} = (A_1|V_o|+A_2)\ (T_s-T_o) \quad \text{over sea, if } T_s>T_o;$$

$$\left(\frac{\delta Q}{dt}\right)_{p_o} = 0 \text{ over land and over ocean areas if } T_s \leq T_o; \tag{4.1.1}$$

where A_1 and A_2 are empirical constants $A_2=10A_1$ $= 5.10^{-3}$ m(sec)$^{-2}$(deg)$^{-1}$. T_s is the sea surface temperature and T_o is the air temperature near the sea surface.

An approximative temperature at the level $p_x = \frac{1}{2}\ (p_o+p_M)$ is obtained through an integration of the hydrostatic equation:

$$T_x = \frac{2f_o}{R \ln\left(\frac{p_o}{p_m}\right)} \psi_1. \tag{4.1.2}$$

We now assume that Γ is constant in the layer and is 75% of Γ_d. We therefore get

$$T_o = T_x+0.75\ \Gamma_d\ \frac{1}{2}(p_o-p_m) = \frac{2f_o}{R \ln\left(\frac{p_o}{p_m}\right)}\left[1 + \frac{0.75\ R(p_o-p_m)}{c_p(p_o+p_m)}\right]\psi_1. \tag{4.1.3}$$

In equations (2.13) and (2.14) the sensible heating (H) was defined to be:

$$H = \frac{R}{4C_p} \ln \left(\frac{Po}{Pm}\right) \left(\frac{\delta Q}{dt}\right)_{Po} .$$

Substitution of $\left(\frac{\delta Q}{dt}\right)_{Po}$ from equation (4.1.1) and using the expression for T_o from equation (4.1.3) gives:

$$H = h_1 \cdot 10^{-2} (0.5 \cdot 10^{-1} |V_o| - 0.5) \ (T_s - h_2 \cdot \psi_1) \tag{4.1.4}$$

where

$$h_1 = \frac{R}{4c_p} \ln \left(\frac{p_o}{p_m}\right) ;$$

$$h_2 = \frac{2f_o}{R \ln \left(\frac{p_o}{p_m}\right)} \left[1 + \frac{0.75 \ R(p_o - p_m)}{c_p (p_o + p_m)} \right] .$$

4.2 PROGNOSTIC EQUATION FOR HUMIDITY

The specific humidity can be predicted by the following equation:

$$\frac{\partial q}{\partial t} = - \mathbb{V} \cdot \nabla q - \omega \frac{\partial q}{\partial p} + \varepsilon - r + A \nabla^2 q . \tag{4.2.1}$$

Here ε denotes evaporation, r condensation, and A is the coefficient of dissipation. We will disregard ε and r will be introduced in a different way. Using the continuity equation we get the following prognostic equation:

$$\frac{\partial q}{\partial t} = - \nabla \cdot (q \ \mathbb{V}) - \frac{\partial}{\partial p} (q\omega) + A \nabla^2 q. \tag{4.2.2}$$

Since our model has a very low vertical resolution we have to parameterize the vertical distribution of humidity. We will, therefore, use precipitable water as the prognostic variable. The precipitable water is defined as

$$P_W = \int_0^{\infty} \rho_W \, dz = \frac{1}{g} \int_{p_T}^{p_o} q \, dp \tag{4.2.3}$$

where

ρ_W = the density of water vapor and p_T is the pressure at the level over which we can disregard the humidity. (Here we have p_T = 30 cb and p_o = 100 cb.)

A new quantity

$$w = \frac{1}{p_o - p_T} P_W \tag{4.2.4}$$

which may be called normalized precipitable water is introduced. We assume that $q(x,y,p,t) = gE(p)w(x,y,t)$ where $E(p)$ describes the vertical variation of q computed from the standard atmosphere under the assumption that the relative humidity is 50%. The expression (4.2.4) is now introduced into (4.2.2) and we then integrate with respect to p from p_o to P_1 to give

$$\frac{\partial w}{\partial t} = - \nabla \cdot (\tilde{\mathbb{V}} w) - w d' \omega_s + A \nabla^2 w \tag{4.2.5}$$

where

$$\tilde{\mathbb{V}} = \frac{1}{p_o - p_T} \int_{p_T}^{p_o} \mathbb{V}(p) \, E(p) \, dp \text{ and } d' = \frac{E(p_o)}{p_o - p_T} .$$

With $\mathbb{V} = \mathbb{k}\times\nabla\psi + \nabla\chi$ equation (4.2.5) can be written

$$\frac{\partial w}{\partial t} = - J(\tilde{\psi},w) - \nabla\tilde{\chi}\cdot\nabla w - w\nabla^2\tilde{\chi} - wd'\omega_s + A\nabla^2 w \qquad (4.2.6)$$

where

$$\tilde{\psi} = e_m\psi_m + e_1\psi_1 + e_2\psi_2 \quad \text{and} \quad \nabla^2\tilde{\chi} = \tilde{D} = e_m D_m + e_1 D_1 + e_2 D_2.$$

4.3 LATENT HEAT

There are two conditions for condensation:

(a) $\bar{\omega}_1 < \omega_{tol}$ (where ω_{tol} is a given tolerance)

(b) the relative humidity should exceed and be equal to 80%.

If these two conditions are valid, the latent heat is computed by the aid of expression (4.3.1)

$$H_{lat} = \frac{R}{4c_p} \ln\left(\frac{p_o}{p_m}\right) \left(\frac{\delta Q}{dt}\right)_{lat} = h_1 \left(\frac{\delta Q}{dt}\right)_{lat}$$

where (4.3.1)

$$\left(\frac{\delta Q}{dt}\right)_{lat} = - L\cdot\bar{\omega}_1 F \quad \text{and} \quad F = \frac{1}{1 + \frac{L}{c_p}\left(\frac{\partial q^x}{\partial t}\right)_p} \left[\left(\frac{\partial q^x}{\partial p}\right)_T + \frac{R}{c_p}\frac{T}{p}\left(\frac{\partial q^x}{\partial T}\right)_p\right]$$

q^x is the maximum specific humidity. If the conditions (a) and (b) are not valid, we put $H_{lat} = 0$. Also see paragraph 8.4.

5. COMPUTATION OF THE VERTICAL MOTION

The integrated vertical motions in the two layers 1 and 2 are computed in the model. With the aid of the equation (2.6) and (2.9) we obtain:

$$\bar{\omega}_1 = \frac{1}{p_o - p_m} \int_{p_m}^{p_o} \omega dp = \frac{p_o + p_m - p_1}{2p_o - p_1} \omega_s + \frac{\frac{1}{3}(2p_1 - 3p_m - p_o)(p_o - p_m)}{2p_o - p_1} D_1$$

$$- \frac{p_m(p_o - p_m)}{2p_o - p_1} D_2; \qquad (5.1)$$

$$\bar{\omega}_1 = t_1 \omega_s + t_2 D_1 - t_3 D_2.$$

In the same way we obtain from the equation (2.7) and (2.9):

$$\bar{\omega}_2 = \frac{1}{p_m - p_1} \int_{p_1}^{p_m} \omega dp = \frac{p_m}{2p_o - p_1} \omega_s - \frac{p_m(p_o - p_m)}{2p_o - p_1} D_1 +$$

$$+ \frac{\frac{1}{3}(2p_o - p_1)(p_m - p_1) - p_m(2p_o - p_m - p_1)}{2p_o - p_1} D_2; \qquad (5.2)$$

$$\bar{\omega}_2 = t_4 \omega_s - t_3 D_1 + t_5 D_2 .$$

Here we have

$$t_1 = \frac{p_o + p_m - p_1}{2p_o - p_1}$$

$$t_2 = \frac{1}{3} \frac{(2p_1 - 3p_m - p_o)(p_o - p_m)}{2p_o - p_1}$$

$$t_3 = \frac{p_m(p_o - p_m)}{2p_o - p_1}$$

$$t_4 = \frac{p_m}{2p_o - p_1}$$

$$t_5 = \frac{\frac{1}{3}(2p_o - p_1)(p_m - p_1) - p_m(2p_o - p_m - p_1)}{(2p_o - p_1)}$$

The physical parameters are computed by the subroutine COEFF3P.

(See appendix B to this report)

6. NUMERICAL VALUES OF THE CONSTANTS

For the levels $p_o = 100$ cb, $p_m = 50$ cb, $p_1 = 30$ cb and the stabilities $(\Gamma_d - \Gamma)_1 = 0.422222$, $(\Gamma_d - \Gamma)_2 = 0.511111$ we get the following numerical values for the constants:

$c_1 = \frac{10}{17} = 0.588235$	$m_1 = -826.330$	$s_1 = 28.230856$
$c_2 = \frac{10}{17} = 0.588235$	$m_2 = -688.40$	$s_2 = 10.645832$
$c_3 = \frac{48}{51} = 0.941176$	$n_1 = -532.92$	$a_3 = 0.044374$
$c_4 = \frac{24}{51} = 0.470588$	$n_2 = -10.58.36$	$b_3 = 0.012258$
$c_5 = \frac{40}{51} = 0.784314$	$a_1 = 2.0828 \cdot 10^{-3}$	$h_1 = 0.495351 \cdot 10^{-1}$
$c_6 = \frac{40}{51} = 0.784314$	$a_2 = 1.3546 \cdot 10^{-3}$	$h_2 = 0.110953 \cdot 10^{-5}$
		$h_3 = 1.03552 \cdot 10^{-6}$
		$h_4 = 0$
		$h_5 = 0.492929 \cdot 10^{-1}$
		$h_6 = 1.03552 \cdot 10^{-6}$
$c_7 = \frac{3}{51} = 0.588235$	$b_1 = 1.0475 \cdot 10^{-3}$	$t_1 = 0.705882$
$c_8 = \frac{2}{170} = 0.0117647$	$b_2 = 1.6261 \cdot 10^{-3}$	$t_2 = -18.627500$
		$t_3 = 14.705900$
		$t_4 = 0.294118$
		$t_5 = -28.627500$

7. INTEGRATION OF THE COMPLETE VORTICITY EQUATION

7.1 GENERAL ASPECTS

Very little knowledge exists about the effect of the small order terms in the vorticity equation: the advection of vorticity by the divergent wind, $\mathbb{V}\chi \cdot \nabla\zeta$; the product of relative vorticity and divergence, $\zeta\nabla \cdot \mathbb{V}$; the vertical advection of vorticity, $\omega\frac{\partial\zeta}{\partial p}$; and the twisting term $\mathbb{k} \cdot (\frac{\partial \mathbb{V}}{\partial p} x \nabla\omega)$.

If these terms are used it is necessary, in order to conserve the total energy for an adiabatic model, to use the complete balance equation. If this is not the case, the model will not conserve total energy and after a certain time the development starts to deteriorate. This judgment has been mainly <u>qualitative</u> and we do not know the size of the error due to this inconsistency. It may be that this error is relatively small in comparison to other errors, as for instance, uncertainties of the initial state, and uncertainties in the description of the dissipation and the heating mechanisms. Therefore, it is necessary to perform a more detailed study of the problem and base our decision on a quantitative investigation.

It is by no means evident that we should use the same kind of assumptions in the formulation of models for short-range predictions (24 hours) as for models for medium- and long-range prediction. An example will illustrate this.

If one is interested in long-time integrations of the barotropic vorticity equation, it is necessary to use a finite difference expression which conserves kinetic energy, vorticity, and mean-squared vorticity. A finite difference expression which conserves these identities has been derived by Arakawa. However, if one is interested in short-range predictions, the so-called "Arakawa Jacobian" is not recommended since the phase-speed error is larger than the conventional finite difference analog to the Jacobian operator which only conserves vorticity. Experiments have shown that for forecasts up to four or five days it is not necessary to use an energy consistent Jacobian operator since the error in the kinetic energy is much smaller than errors due to other effects.

One of the problems which we have with the simplified vorticity equation is the over-prediction of anticyclogenesis. This seems due mainly to the lack of the term $\zeta \nabla \cdot \mathbb{V}$ in the vorticity equation:

$$\frac{\partial \zeta}{\partial t} = \cdots\cdot -(f+\zeta)\nabla \cdot \mathbb{V}. \tag{7.1.1}$$

In areas of convergence, relative vorticity is mostly positive, or will be after a short time. This means that the relative vorticity will increase faster in such areas if the complete expression (7.1.1) is included. On the other hand, in areas of divergence, the relative vorticity is mostly negative, or

will be after some hours. If the complete expression is used, the relative vorticity will decrease more slowly than if we use the simplified expression. It is easily seen that this term will create an asymmetry in the vorticity pattern which is also observed in reality.

Also the vertical advection of vorticity seems to play an important role, especially in cyclone development. During the development of the cyclone the activity is mainly concentrated in two different areas.

One area is in front of the warm front or, later in the development, the occluded part of the front. The other area is found below the upper-air low or trough, where a special center of activity is created in the later stages of the cyclone development. During the development of the cyclone, an area of sinking motion is concentrated under the upper-air low.

$$\frac{\partial \zeta}{\partial t} = -\omega \frac{\partial \zeta}{\partial p} \qquad (7.1.2)$$

It is easily seen from the vorticity equation (7.1.2) that this effect will give an increase in the relative vorticity in areas of sinking motion and where the vorticity increases with height.

Upward motion over a surface low yields, in the same way, an increase in the vorticity for the levels above. Therefore the vertical advection of vorticity will <u>increase the speed of occlusion</u>. Figure 2 shows two different 12-hour

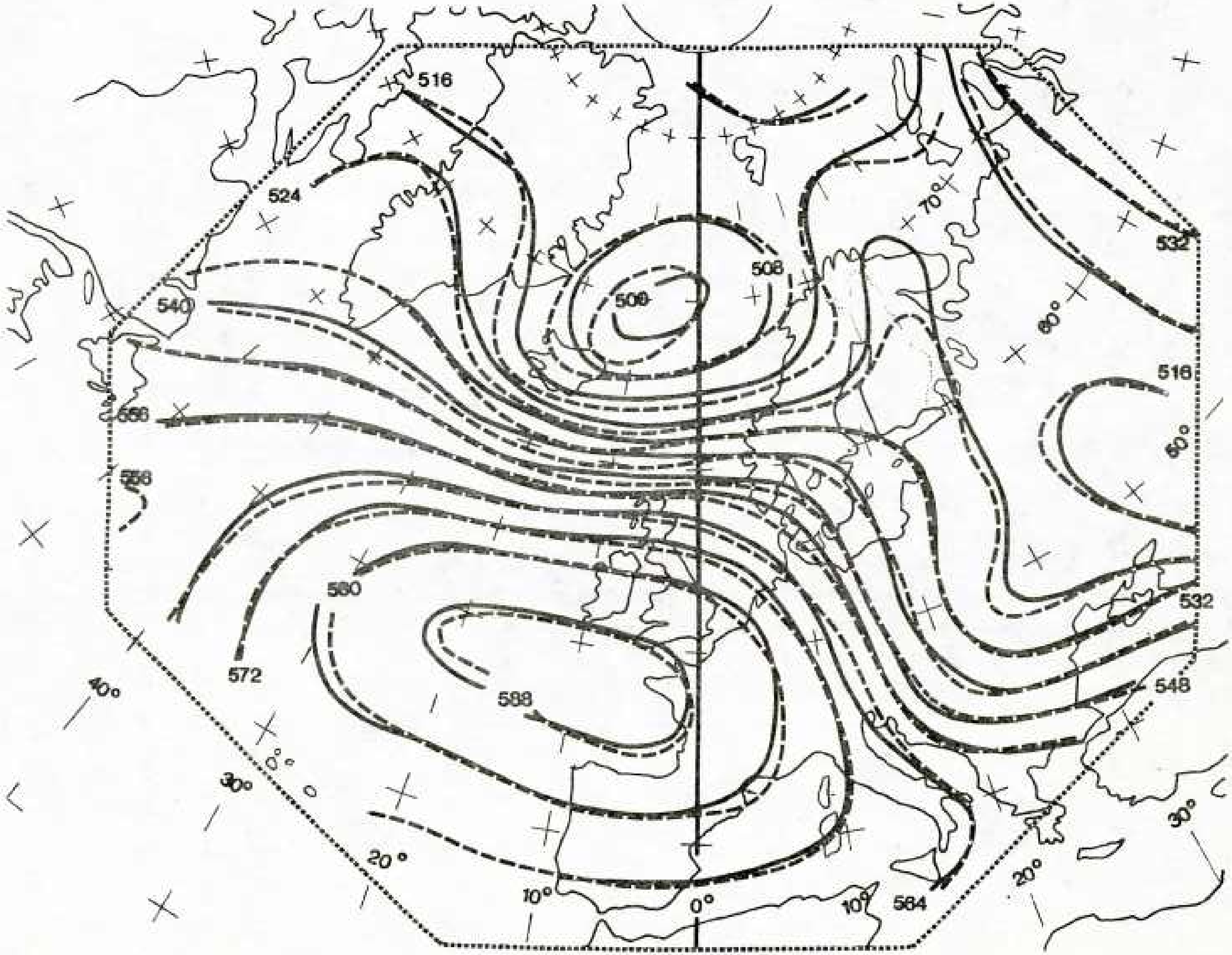


Figure 2. Two example 12-hour predictions with a 5-layer quasi-geostrophic model.

predictions with a 5-layer quasi-geostrophic model. Solid lines indicate a prediction performed with a quasi-geostrophic energy consistent model. Dashed lines indicate a prediction where the terms $-(\zeta\nabla\cdot \mathbb{V} + \omega\frac{\partial\zeta}{\partial p})$ have been included.

7.2 DERIVATION OF THE FORECASTING EQUATIONS

The complete vorticity equation reads:

$$\frac{\partial\zeta}{\partial t} = - \mathbb{V}_\psi\cdot\nabla\eta \underbrace{- \mathbb{V}_\chi\cdot\nabla\eta}_{1} - f\nabla\cdot\mathbb{V}_\chi \underbrace{- \zeta\,\nabla\cdot\mathbb{V}_\chi}_{2} \underbrace{- \omega\frac{\partial\zeta}{\partial p}}_{3} \underbrace{+ \mathbb{k}\cdot(\frac{\partial\mathbb{V}_\psi}{\partial p}x\nabla\omega)}_{4}. \quad (7.2.1)$$

Here V_ψ is the non-divergent wind and V_χ the divergent wind. The terms 1, 2, 3 and 4 are denoted non-geostrophic terms. Integrating through layer 1 by the representation of V, ζ and D given in section 2 yields

$$(p_o-p_m)\ \left[\frac{\partial\zeta_m}{\partial t} - \frac{\partial\zeta_1}{\partial t}\right] = (p_o-p_m)\left[- \mathbb{V}_m\cdot\nabla(\zeta_m-\zeta_1+f) + \mathbb{V}_1\cdot\nabla(\zeta_m+f)\right.$$
$$\left.- \frac{4}{3}\mathbb{V}_1\cdot\nabla\zeta_1 - f(D_m-D_1) - \zeta_m(D_m-D_1) + \zeta_1(D_m - \frac{4}{3}D_1)\right]$$
$$+2\zeta_1\bar{\omega}_1 - 2\,\mathbb{k}\cdot(\mathbb{V}_1x\nabla\bar{\omega}_1) \quad (7.2.2)$$

Integration through layer 2 and layer 3 yields in a similar way

$$(p_m-p_1)\ \left[\frac{\partial\zeta_m}{\partial t} + \frac{\partial\zeta_2}{\partial t}\right] = (p_m-p_1)\left[- \mathbb{V}_m\cdot\nabla(\zeta_m+\zeta_2+f) - \mathbb{V}_2\cdot\nabla(\zeta_m+f)\right.$$
$$\left.- \frac{4}{3}\mathbb{V}_2\cdot\nabla\zeta_2 - f(D_m+D_2) - \zeta_m(D_m+D_2) - \zeta_2(D_m+\frac{4}{3}D_2)\right]$$
$$+ 2\zeta_2\bar{\omega}_2 - 2\,\mathbb{k}\cdot(\mathbb{V}_2x\nabla\bar{\omega}_2) \quad (7.2.3)$$

$$\frac{1}{2}p_1\left[\frac{\partial\zeta_m}{\partial t}+2\frac{\partial\zeta_2}{\partial t}\right]=\frac{1}{2}p_1\left[-(\mathbb{V}_m+2\mathbb{V}_2)\cdot\nabla f-\frac{2}{3}(\mathbb{V}_m+2\mathbb{V}_2)\cdot\nabla(\zeta_m+2\zeta_2)\right.$$

$$\left.-f(D_m+2D_2)-\frac{2}{3}(\zeta_m+2\zeta_2)(D_m+2D_2)\right]-(\zeta_m+2\zeta_2)\bar{\omega}_3$$

$$+\mathbb{k}\cdot\left\{\mathbb{V}_m+2\mathbb{V}_2)\times\nabla\bar{\omega}_3\right\} \tag{7.2.4}$$

where $\bar{\omega}_1$ and $\bar{\omega}_2$ are given in (5.1) and (5.2) and $\bar{\omega}_3$ is given by:

$$\bar{\omega}_3=\frac{1}{p_1}\int_o^{p_1}\omega dp. \tag{7.2.5}$$

If we now add (7.2.2), (7.2.3) and (7.2.4) and divide by $\frac{2p_o-p_1}{2}$, we get a prognostic equation for the vertically integrated mean vorticity. From now on we will only keep the non-geostrophic terms on the right hand side of the equation.

$$\frac{\partial}{\partial t}(\zeta_m+c_1\zeta_2-c_2\zeta_1)=\{-(\mathbb{V}_\chi)_m\cdot\nabla(c_3\eta_m-c_2\zeta_1+c_4\zeta_2+c_7f)$$

$$-(\mathbb{V}_\chi)_1\cdot\nabla(-c_2\zeta_m+c_5\zeta_1)-(\mathbb{V}_\chi)_2\cdot\nabla(c_4\eta_m+c_6\zeta_2+2c_7f)\}_1$$

$$+\{c_8\zeta_m\omega_s+c_2\zeta_1(D_m-\frac{4}{3}D_1)-\zeta_2(c_4D_m+c_6D_2)+\zeta_m(c_7D_m+2c_7D_2)\}_2$$

$$+c_8\{2\zeta_1\bar{\omega}_1+2\zeta_2\bar{\omega}_2-(\zeta_m+2\zeta_2)\bar{\omega}_3\}_3+c_8\{\mathbb{k}\cdot[-2\mathbb{V}_1\times\nabla\bar{\omega}_1-2\mathbb{V}_2\times\nabla\bar{\omega}_2$$

$$+(\mathbb{V}_m+2\mathbb{V}_2)\times\nabla\bar{\omega}_3]\}_4 \tag{7.2.6}$$

The two remaining prognostic equations are computed from the difference between the vorticity equation for p_m and p_o and the difference between the vorticity equation for p_1 and p_m. We will then have two vorticity equations valid for layer 1 and layer 2 respectively. The geostrophic terms are omitted.

$$\frac{\partial \zeta_1}{\partial t} = - \{((\mathbb{V}_\chi)_m - (\mathbb{V}_\chi)_1) \cdot \nabla \zeta_1 + (\mathbb{V}_\chi)_1 \cdot \nabla (\eta_m - \zeta_1)\}_1$$

$$-\{(\zeta_m - 2\zeta_1) D_1 + \zeta_1 D_m\}_2 + \{\frac{\zeta_1}{p_o - p_m} (\omega_m - \omega_s)\}_3$$

$$-\{\frac{1}{(p_o - p_m} \mathbb{k} \cdot [(\mathbb{V}_\psi)_1 x \nabla (\omega_m - \omega_s)]\}_4 \qquad (7.2.7)$$

$$\frac{\partial \zeta_2}{\partial t} = - \{((\mathbb{V}_x)_m + (\mathbb{V}_x)_2) \cdot \nabla \zeta_2 + (\mathbb{V}_x)_2 \cdot \nabla (\eta_m + \zeta_2)\}_1$$

$$- \{(\zeta_m + 2\zeta_2) D_2 + \zeta_2 D_m\}_2 + \{\frac{1}{p_m - p_1} \zeta_2 (\omega_1 - \omega_m)\}_3$$

$$- \{\frac{1}{p_m - p_1} \mathbb{k} \cdot [(\mathbb{V}_\psi)_2 x \nabla (\omega_1 - \omega_m)]\}_4 \qquad (7.2.8)$$

Subscript 1 indicates contribution from $\mathbb{V}_\chi \cdot \nabla \eta$

Subscript 2 indicates contribution from $\zeta \cdot \nabla \mathbb{V}$

Subscript 3 indicates contribution from $\omega \frac{\partial \zeta}{\partial p}$

Subscript 4 indicates contribution from $\mathbb{k} \cdot (\frac{\partial \mathbb{V}}{\partial p} x \nabla \omega)$

ω_m and ω_1 are the vertical motion at level p_m and p_1 respectively.

The non-geostrophic terms will be approximated by the divergence computed from the geostrophic part of the equations for the preceeding time step.

We can now express the forecasting equations in the following formal way (compare (2.2, 2.13 and 2.14).

$$\nabla^2 \{\frac{\partial}{\partial t} (\psi_m + c_1 \psi_2 - c_2 \psi_1)\} = F_{mG}$$

$$+ F_{m1} + F_{m2} + F_{m3} + F_{m4} \qquad (7.2.9)$$

$$\nabla^2 \frac{\partial \psi_1}{\partial t} - a_1 f^2 \frac{\partial \psi_1}{\partial t} + a_2 f^2 \frac{\partial \psi_2}{\partial t} = F_{1G}$$

$$+ F_{11} + F_{12} + F_{13} + F_{14} \tag{7.2.10}$$

$$\nabla^2 \frac{\partial \psi_2}{\partial t} - b_2 f^2 \frac{\partial \psi_2}{\partial t} + b_1 f^2 \frac{\partial \psi_1}{\partial t} = F_{2G} + F_{21} + F_{22} + F_{23} + F_{24} \tag{7.2.11}$$

F_{mG}, F_{1G} and F_{2G} are the geostrophic and non-adiabatic terms. They correspond to the right hand part of the equations 2.2, 2.13 and 2.14.

The non-geostrophic terms are the same as in the equations (7.2.6), (7.2.7) and (7.2.8).

8. NUMERICAL SOLUTION OF THE 3-PARAMETER MODEL

8.1 GENERAL ASPECTS

The equation of the model will be applied on a polar-stereographic projection. The polar-stereographic plane is assumed to cut the sphere at a latitude ϕ_o. In the numerical computations ϕ_o is put equal to 60°N, however, other values of ϕ_o can easily be chosen. The grid-distance d can also be chosen arbitrarily. For further details see the program description in Appendix B.

The computational area can have any form, from an irregular octagon to a square. The only geometrical condition is that the inner angles of the area must be 90° or 135°.

The coordinate axis of the grid is positively oriented (see Figure 3). The computational area is specified by the coordinates of the corner points ($x_1/y_1 \cdots x_8/y_8$) and by the coordinates of the north pole (x_{pole}/y_{pole}).

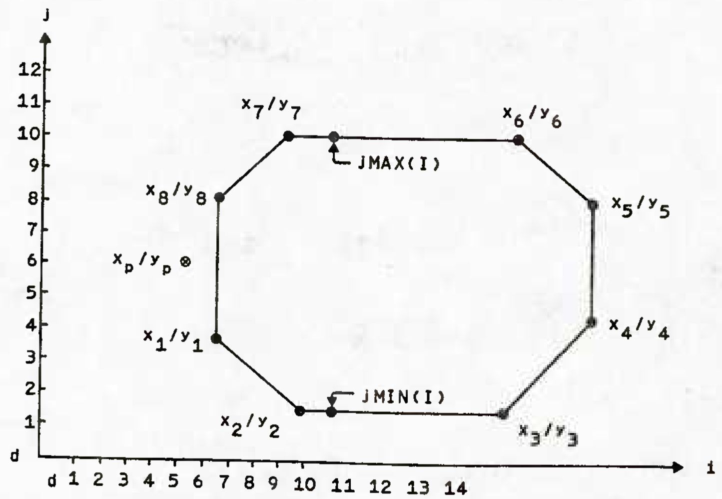


Figure 3. Example computational area.

If the area is reduced to a rectangle $x_1 = x_2$, $y_1 = y_2$, $x_3 = x_4$, $y_3 = y_4$, etc.

8.2 FINITE-DIFFERENCES

In order to transform the differential equations to finite-difference equations we will introduce the following finite-difference notation and operators (α and β are arbitrary quantities):

$$x \simeq i\Delta s$$

$$y \simeq j\Delta s$$

$$t \simeq \tau \Delta t$$

$$f(x,y,t) \simeq f^{\tau}_{i,j}$$

$$\nabla^2\alpha \simeq \frac{1}{d^2} \mathbb{V}^2\alpha \tag{8.2.1a}$$

$$J(\alpha,\beta) \simeq \frac{1}{4d^2} \mathbb{J}(\alpha,\beta) \tag{8.2.1b}$$

$$\frac{\partial\alpha}{\partial t} \simeq \frac{\Delta^{\tau}\alpha}{\varepsilon\Delta t} \tag{8.2.1c}$$

$$(\mathbb{V}^2\alpha)_{ij} = \alpha_{i+1,j} + \alpha_{i-1,j} + \alpha_{i,j+1} + \alpha_{i,j-1} - 4\alpha_{ij} \tag{8.2.2a}$$

$$\mathbb{J}(\alpha;\beta) = (\alpha_{i+1} - \alpha_{i-1})_j(\beta_{j+1} - \beta_{j-1})_i$$

$$- (\alpha_{j+1} - \alpha_{j-1})_i(\beta_{i+1} - \beta_{i-1})_j \tag{8.2.2b}$$

$\tau = 0$ yields $\alpha^{\frac{1}{2}} - \alpha^{0}$ and $\varepsilon = \frac{1}{2}$

$\tau = \frac{1}{2}$ yields $\alpha^{1} - \alpha^{0}$ and $\varepsilon = 1$

$\tau \geqslant 1$ yields $\alpha^{t+1} - \alpha^{t-1}$ and $\varepsilon = 2$ (8.2.2c)

Introducing the map-scale factor $m = \frac{1+\sin\phi_o}{1+\sin\phi}$ and inserting the quantity $\mu = (\frac{m}{d})^2$ we get:

$$\nabla^2(\Delta\psi_1) - a_1\frac{f^2}{\mu}(\Delta\psi_2) + a_2\frac{f^2}{\mu}(\Delta\psi_2) = F_{1G}, \tag{8.2.3}$$

$$\nabla^2(\Delta\psi_2) - b_2\frac{f^2}{\mu}(\Delta\psi_2) + b_1\frac{f^2}{\mu}(\Delta\psi_1) = F_{2G}, \tag{8.2.4}$$

$$\nabla^2(\Delta\psi_M) - \frac{q}{\mu}(\Delta\psi_M) = F_{mG}. \tag{8.2.5}$$

q is an empirical constant to adjust for the very long waves, $q = 0.75\cdot 10^{-12} m^{-2}$.

Here we have (for simplicity we from now on put $\mathfrak{J} = J$):

$$F_{1G} = F_1' + F_1'',$$

$$F_{2G} = F_2' + F_2'',$$

$$F_1' = -\varepsilon\Delta t\cdot\frac{f}{\mu}(a_3\omega_s + a_1 H),$$

$$F_1'' = -\varepsilon\Delta t\frac{1}{4}[J_1 + J_2 + f^2(a_2 J_4 - a_1 J_3)],$$

$$F_2' = -\varepsilon\Delta t\frac{f}{\mu}(b_3\omega_s + b_1 H),$$

$$F_2'' = -\varepsilon\Delta t\frac{1}{4}[J_5 + J_6 + f^2(b_1 J_3 - b_2 J_4)],$$

$$F_{mG} = -\varepsilon\Delta t\frac{1}{4}[J_7 + J_8 + J_9 - 4\frac{f}{\mu}c_8\omega_s].$$

We further have:

$$\beta_2 = \eta_m - 2\zeta_1,$$

$$\beta_6 = \eta_m + 2\zeta_2,$$

$$\beta_7 = c_3\eta_m - c_2\zeta_1 + c_4\zeta_2 + c_7 f,$$

$$\beta_8 = -c_2\eta_m + c_5\zeta_1,$$

$$\beta_9 = c_4\eta_m + c_6\zeta_2 + 2c_7 f,$$

$$J_1 = J(\psi_m;\zeta_1),$$

$$J_2 = J(\psi_1;\beta_2),$$

$$J_3 = J(\psi_m;\psi_1),$$

$$J_4 = J(\psi_m;\psi_2),$$

$$J_5 = J(\psi_m;\zeta_2),$$

$$J_6 = J(\psi_2;\beta_6),$$

$$J_7 = J(\psi_m;\beta_7),$$

$$J_8 = J(\psi_1;\beta_8),$$

$$J_9 = J(\psi_2;\beta_9),$$

$$J_{10} = J(\psi_m-2\psi_1;p_s),$$

$$\omega_s = \frac{\mu}{4} J_{10}-c_f(\zeta_m-2\zeta_1).$$

c_f is an empirical constant and a function of the exchange coefficient of momentum in the boundary layer.

8.3 COMPUTATION OF SENSIBLE HEAT

Equation (4.1.4) reads in finite-difference form:

$$H_{SENS} = 0.5\cdot 10^{-2}\ h\ (0.1\sqrt{\frac{\mu}{4}((\Delta_x\psi_o)^2+(\Delta_y\psi_o)^2)}\ +1)(T_s-h_2\ \psi_1)$$

over ocean, if $(T_s-h_2\psi_1)>0$, (8.3.1)

$H_{SENS} = 0$ over land and if $(T_s-h_2\psi_1) <0$.

$$\Delta_x\psi_o = [\psi_m(i+1)-2\psi_1(i+1)]-[\psi_m(i-1)-2\psi_1(i-1)]$$

$$\Delta_y\psi_o = [\psi_m(j+1)-2\psi_1(j+1)]-[\psi_m(j-1)-2\psi_1(j-1)]$$

8.4 COMPUTATION OF LATENT HEAT (See Gambo 1963)

The latent heat is introduced in the model by the expression:

$$H_{LAT} = -hL\omega^* F^* \qquad \text{if } \bar{\omega}_1 \leqslant -\delta_1$$

$$H_{LAT} = 0 \qquad \text{if } \bar{\omega}_1 > -\delta_1 \tag{8.4.1}$$

$$H_{LAT} = 0 \qquad \text{if } \frac{r}{\varepsilon \Delta t} < \text{tolerance}$$

$$\omega^* = -(\delta_1+\delta_2) \text{ if } \bar{\omega}_1 < -(\delta_1+\delta_2)$$

$$\omega^* = -\left| \frac{\bar{\omega}_1^{\,2}}{\delta_1+\delta_2} \right| \text{ if } -(\delta_1+\delta_2) \leqslant \bar{\omega}_1 \leqslant -\delta_1 \tag{8.4.2}$$

δ_1 and δ_2 are here two tolerances with the same dimension as $\bar{\omega}_1$, r is the precipitation, and δ_2 is introduced for operational purposes.

$$F^* = F^*(p,T) = \frac{\varepsilon \frac{T}{p} E(T) \left[\frac{\varepsilon L}{c_p} - T \right]}{pT^2 + \frac{\varepsilon^2 L^2}{C_p R} E(T)} \tag{8.4.3}$$

$$E(T) = E_o e^{\frac{\varepsilon L}{R}\left(\frac{1}{T_o} - \frac{1}{T}\right)}$$

$\varepsilon = 0.622$

$L = 2.5 \cdot 10^6$ $\qquad P = P_{mean} = \frac{P_m + P_o}{2}$

$C_p = 1004$

$T_o = 273$

$E_o = 0.611$ $\qquad T = \frac{2f_o}{R \ln\left(\frac{P_o}{P_m}\right)} \psi_1 = h_6 \psi_1$

8.4.1 Computation of $\tilde{\mathbb{V}}$ and $\tilde{D}$

We will compute the humidity in the layer $P_o = 100$ cb and $P_T = 30$ cb ; defining $\tilde{\mathbb{V}}$ as

$$\tilde{\mathbb{V}} = \frac{1}{70} \int_{30}^{100} \mathbb{V}(P)\ E(P)\ dp$$

and inserting the wind profile yields

$$\tilde{\mathbb{V}} = e_m \mathbb{V}_m + e_1 \mathbb{V}_1 + e_2 \mathbb{V}_2 \tag{8.4.4}$$

where

$$e_m = \frac{1}{70}[15E(100)a_{100} + 25E(70)a_{70} + 20E(50)a_{50} + 10E(30)a_{30}],$$

$$e_1 = \frac{1}{70}[15E(100)b_{100} + 25E(70)b_{70} + 20E(50)b_{50} + 10E(30)b_{30}],$$

$$e_2 = \frac{1}{70}[15E(100)c_{100} + 25E(70)c_{70} + 20E(50)c_{50} + 10E(30)c_{30}]. \tag{8.4.5}$$

The constants a_p, b_p and c_p, where p assumes the values 100, 70, 50, 30 are computed for the three following alternatives:

I	II	III
$p_m \leq p \leq p_o$	$p_1 \leq p < p_m$	$0 \leq p < p_1$
$a_p = 1$	$a_p = 1$	$a_p = \frac{p}{p_1}$
$b_p = -\frac{2(p-p_m)}{p_o-p_m}$	$b_p = 0$	$b_p = 0$
$c_p = 0$	$c_p = \frac{2(p_m-p)}{p_m-p_1}$	$c_p = \frac{2p}{p_1}$

E(100)=2.5415, E(70)=0.9683, E(50)=0.3527, E(30)=0.0613

Correspondingly we get

$$\tilde{D} = e_m D_m + e_1 D_1 + e_2 D_2 = (e_1 + e_m c_2) D_1 + (e_2 - e_m c_1) D_2 - e_m c_8 w_s$$

8.4.2 Computation of Precipitation

Equation (4.2.6) reads in finite-difference form (the term $\nabla\tilde{\chi}\nabla w$ disregarded and w replaced with q):

$$q^{\tau+1} = q^{\tau-1} + \varepsilon\Delta t\, H_q^{\tau},$$

$$H_q^{\tau} = -\frac{\mu}{4} J(\tilde{\psi}^{\tau}; q^{t}) - q^{\tau}(\tilde{D}^{\tau} + dw_s^{\tau}) + A_{diff}\mu\nabla^2 q^{\tau}. \qquad (8.4.6)$$

The precipitation is computed in the following way (precipitation is indicated by r):

$$q_{SAT} = q(\bar{T}_{\psi}) = q(h_3\psi_1 + h_4\psi_2)$$

If $(q^{\tau+1} - 0.8\, q_{SAT}) > 0$ then $q^{\tau+1} = 0.8\, q_{SAT}$,

$$r = \Delta p(q^{\tau+1} - 0.8 q_{SAT}).$$

If $(q^{\tau+1} - 0.8\, q_{SAT}) < 0$ then $r = 0$ and if

$$(q^{\tau+1} - 0.2 q_{SAT}) < 0 \text{ then } q_{mod}^{\tau+1} = 0.2\, q_{SAT}. \qquad (8.4.7)$$

r is accumulated for every timestep and printed out at certain prescribed times. See Appendix B.

8.5 NUMERICAL SOLUTION OF THE NON-GEOSTROPHIC TERMS

The finite difference analogues for (7.2.9), (7.2.10) and (7.2.11) read:

$$\nabla^2 (\Delta\psi_M) - \frac{q}{\mu}(\Delta\psi_M) = F_{mG} + \frac{\varepsilon\Delta t}{\mu} \sum_{i=1}^{4} F_{mi} \tag{8.5.1}$$

$$\nabla^2 (\Delta\psi_1) - a_1\frac{f^2}{\mu}(\Delta\psi_1) + a_2\frac{f^2}{\mu}(\Delta\psi_2) = F_{1G} + \frac{\varepsilon\Delta t}{\mu} \sum_{i=1}^{4} F_{1i} \tag{8.5.2}$$

$$\nabla^2 (\Delta\psi_2) - b_2\frac{f^2}{\mu}(\Delta\psi_2) + b_1\frac{f^2}{\mu}(\Delta\psi_1) = F_{2G} + \frac{\varepsilon\Delta t}{\mu} \sum_{i=1}^{4} F_{2i} \tag{8.5.3}$$

$$\begin{aligned} F_{m1} = -\frac{\mu}{2}\{ & \nabla\chi_m \cdot \nabla \underbrace{[c_3(\zeta_m+f) - c_2\zeta_1 + c_4\zeta_2 + c_7 f]}_{\beta_7} \\ & + \nabla\chi_1 \cdot \nabla \underbrace{[-c_2(\zeta_m+f) + c_5\zeta_1]}_{\beta_8} \\ & + \nabla\chi_2 \cdot \nabla \underbrace{[c_4(\zeta_m+f) + c_6\zeta_2 + 2c_7 f]}_{\beta_9}\} \end{aligned} \tag{8.5.4a}$$

$$\begin{aligned} F_{m2} = [& D_1(k_1\zeta_1 + k_2\zeta_2 + k_3\zeta_m) + D_2(k_4\zeta_1 + k_5\zeta_2 + k_6\zeta_m) \\ & + \omega_s(k_7\zeta_1 + k_8\zeta_2 + k_9\zeta_m)] \end{aligned} \tag{8.5.4b}$$

$$\begin{aligned} F_{m3} = [& D_1(k_{10}\zeta_1 + k_{11}\zeta_2 + k_{12}\zeta_m) + D_2(k_{13}\zeta_1 + k_{14}\zeta_2 + k_{15}\zeta_m) \\ & + \omega_s(k_{16}\zeta_1 + k_{17}\zeta_2 + k_{18}\zeta_m)] \end{aligned} \tag{8.5.4c}$$

$$F_{m4} = \frac{\mu}{2}\{\nabla\psi_1 \cdot \nabla \underbrace{[k_{19}D_1 + k_{20}D_2 + k_{21}\omega_s]}_{\gamma_7}$$

$$+ \nabla\psi_2 \cdot \nabla \underbrace{[k_{22}D_1 + k_{23}D_2 + k_{24}\omega_s]}_{\gamma_8}$$

$$+ \nabla\psi_m \cdot \nabla \underbrace{[k_{25}D_1 + k_{26}D_2 + k_{27}\omega_s]}_{\gamma_9}\} \tag{8.5.4d}$$

$$F_{11} = -\frac{\mu}{2}\{\nabla\chi_m \cdot \nabla\zeta_1 + \nabla\chi_1 \cdot \nabla(\zeta_m + f - 2\zeta_1)\} \tag{8.5.5a}$$

$$F_{12} = [D_1(k_{28}\zeta_1 - \zeta_m) + D_2 k_{29}\zeta_1 + \omega_s k_{30}\zeta_1] \tag{8.5.5b}$$

$$F_{13} = [\zeta_1(k_{31}D_1 + k_{32}D_2 + k_{33}\omega_s)] \tag{8.5.5c}$$

$$F_{14} = \frac{\mu}{2}\{\nabla\psi_1 \cdot \nabla \underbrace{[k_{31}D_1 + k_{32}D_2 + k_{33}\omega_s]}_{\gamma_{10}}\} \tag{8.5.5d}$$

$$F_{21} = -\frac{\mu}{2}\{\nabla\chi_m \cdot \nabla\zeta_2 + \nabla\chi_2 \cdot \nabla(\zeta_m + f + 2\zeta_2)\} \tag{8.5.6a}$$

$$F_{22} = \{D_1 k_{34}\zeta_2 + D_2[k_{35}\zeta_2 - \zeta_m] + \omega_s k_{30}\zeta_2\} \tag{8.5.6b}$$

$$F_{23} = [\zeta_2(k_{36}D_1 + k_{37}D_2 + k_{38}\omega_s)] \tag{8.5.6c}$$

$$F_{24} = \frac{\mu}{2}\{\nabla\psi_2 \cdot \nabla \underbrace{[k_{36}D_1 + k_{37}D_2 + k_{38}\omega_s]}_{\gamma_{11}} \tag{8.5.6d}$$

The computation of the non-geostrophic forcing function

$$\frac{\varepsilon\Delta t}{\mu}\sum_{i=1}^{4} F_{mi}, \quad \frac{\varepsilon\Delta t}{\mu}\sum_{i=1}^{4} F_{1i} \quad \text{and} \quad \frac{\varepsilon\Delta t}{\mu}\sum_{i=1}^{4} F_{2i}$$

are performed by a separate program. See Appendix B.

8.5.1 Numerical Coefficients for the Non-geostrophic Terms

Expressions for constants of the non-geostrophic terms read:

$$\bar{\omega}_3 = t_6\omega_s + t_7 D_1 + t_8 D_2$$

$$\omega_m = t_9\omega_s + t_{10} D_1 + t_{11} D_2$$

$$\omega_1 = t_{12}\omega_s + t_{13} D_1 + t_{14} D_2$$

$$t_6 = c_8 \frac{p_1}{6}$$

$$t_7 = -c_2 \frac{p_1}{6}$$

$$t_8 = (c_1-2) \frac{p_1}{6}$$

$$t_9 = 1 - c_8(p_o - p_m)$$

$$t_{10} = (c_2-1)(p_o-p_m)$$

$$t_{11} = -c_1(p_o-p_m)$$

$$t_{12} = c_8 \frac{p_1}{2}$$

$$t_{13} = -c_2 \frac{p_1}{2}$$

$$t_{14} = (c_1-2) \frac{p_1}{2}$$

$$k_1 = c_2(c_2-\tfrac{4}{3})$$

$$k_2 = -c_2 c_4$$

$$k_3 = c_2 c_7$$

$$k_4 = -c_1c_2$$

$$k_5 = c_1c_4-c_6$$

$$k_6 = c_7(2-c_1)$$

$$k_7 = -c_2c_8$$

$$k_8 = c_4c_8$$

$$k_9 = c_8(1-c_7)$$

$$k_{10} = 2t_2c_8$$

$$k_{11} = -2(t_3+t_7)c_8$$

$$k_{12} = -t_7c_8$$

$$k_{13} = -2t_3c_8$$

$$k_{14} = 2(t_5-t_8)c_8$$

$$k_{15} = -t_8c_8$$

$$k_{16} = 2t_1c_8$$

$$k_{17} = 2(t_4-t_6)c_8$$

$$k_{18} = -t_6c_8$$

$$k_{19} = 2t_2c_8$$

$$k_{20} = -2t_3c_8$$

$$k_{21} = 2t_1c_8$$

$$k_{22} = -\ 2(t_3+t_7)c_8$$

$$k_{23} = 2(t_5-t_8)c_8$$

$$k_{24} = 2(t_4-t_6)c_8$$

$$k_{25} = -t_7c_8$$

$$k_{26} = -t_8c_8$$

$$k_{27} = -t_6c_8$$

$$k_{28} = 2-c_2$$

$$k_{29} = +c_1$$

$$k_{30} = +c_8$$

$$k_{31} = -\frac{t_{10}}{(p_o-p_m)}$$

$$k_{32} = -\frac{t_{11}}{(p_o-p_m)}$$

$$k_{33} = \frac{1-t_9}{(p_o-p_m)}$$

$$k_{34} = -c_2$$

$$k_{35} = c_1-2$$

$$k_{36} = \frac{t_{10}-t_{13}}{(p_m-p_1)}$$

$$k_{37} = \frac{t_{11}-t_{14}}{(p_m-p_1)}$$

$$k_{38} = \frac{t_9-t_{12}}{(p_m-p_1)}$$

For p_o=100, p_m=50 and p_1=30 we have the following numerical values for the constants:

t_6 = 0.0588235

t_7 = -2.941175

t_8 = -7.058825

t_9 = 0.411765

t_{10} = -20.588250

t_{11} = -29.411750

t_{12} = 0.176471

t_{13} = -8.823525

t_{14} = -21.176475

k_1 = -0.438291

k_2 = -0.276816

k_3 = 0.034602

k_4 = -0.346020

k_5 = -0.507498

k_6 = 0.083045

k_7 = -0.006920

k_8 = 0.005536

k_9 = 0.011073

k_{10} = -0.438294

k_{11} = -0.276817

$k_{12} = 0.034602$

$k_{13} = -0.346021$

$k_{14} = -0.507498$

$k_{15} = 0.083045$

$k_{16} = 0.016609$

$k_{17} = 0.005536$

$k_{18} = -0.000692$

$k_{19} = -0.438293$

$k_{20} = -0.346021$

$k_{21} = 0.016609$

$k_{22} = -0.276817$

$k_{23} = -0.507497$

$k_{24} = 0.005536$

$k_{25} = 0.034602$

$k_{26} = 0.083045$

$k_{27} = -0.000692$

$k_{28} = 1.411765$

$k_{29} = +0.588235$

$k_{30} = +0.0117647$

$k_{31} = 0.411765$

$k_{32} = 0.588235$

$k_{33} = 0.0117647$

$k_{34} = -0.588235$

$k_{35} = -1.411765$

$k_{36} = -0.588235$

$k_{37} = -0.411765$

$k_{38} = 0.0117647$

9. INITIALIZATION

The stream functions ψ are computed from the geopotential Z by the relation

$$g\nabla^2 Z = \nabla\cdot(f\nabla\psi) \quad \text{or}$$

$$\nabla^2\psi = \frac{g}{f}\nabla^2 Z - \frac{1}{f}\nabla f\cdot\nabla\psi; \tag{9.1}$$

inserting

$$\nabla\psi = \frac{g}{f}\nabla Z \qquad \text{yields}$$

$$\nabla^2\psi = \frac{g}{f}\nabla^2 Z - \frac{g}{f^2}\nabla f\cdot\nabla Z. \tag{9.2}$$

The boundary values of ψ are computed by the relation (γ is a constant)

$$\frac{\partial\psi}{\partial s} = \frac{g}{f}\frac{\partial Z}{\partial s} + \gamma. \tag{9.3}$$

If we assume that there is no net flow out of the area it is easily seen that $\gamma = -\frac{1}{L}\oint \frac{g}{f}\frac{\partial Z}{\partial s}ds$. The integration is performed along the boundary of the area in positive order. The integration starts in point x_1/y_1 where we put

$$\psi(x_1;y_1) = \frac{g}{f(x_1;y_1)} Z(x_1;y_1). \tag{9.4}$$

Z is computed from ψ in a similar way using the following equation:

$$\nabla^2 Z = \frac{f}{g}\nabla^2\psi + \frac{1}{g}\nabla f\cdot\nabla\psi. \tag{9.5}$$

Boundary values for Z are computed initially and stored in a special string. See Appendix B.

9.1 NUMERICAL SOLUTION

Equation (9.2) reads in finite difference form:

$$\nabla^2\psi = \frac{g}{f}\nabla^2 Z - \frac{g}{2f^2}\nabla f\cdot\nabla Z \qquad (9.6)$$

where the standard five point formulas are used to compute $\nabla^2\psi$, $\nabla^2 Z$ and $\nabla f \nabla Z$ is defined as

$$\nabla f\cdot\nabla Z = (f_{i+1}-f_i)_j(Z_{i+1}-Z_i)_j + (f_i-f_{i-1})_j(Z_i-Z_{i-1})_j + (f_{j+1}-f_j)_i(Z_{j+1}-Z_j)_i + (f_j-f_{j-1})_i(Z_j-Z_{j-1})_i .$$

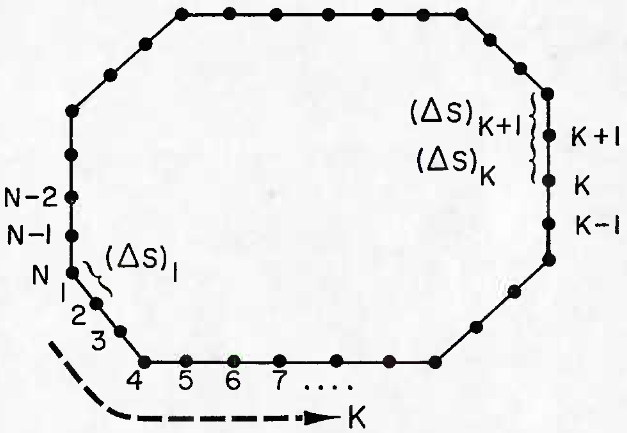
Equation (9.3) reads in finite difference form:

$$\psi_{k+1} = \psi_k + 2g\,\frac{Z_{k+1}-Z_k}{f_{k+1}+f_k} + \gamma(\Delta s)_k \qquad (9.7)$$

where

$$\gamma = -\frac{1}{L}\,2g\sum_{k=1}^{N-1}\frac{Z_{k+1}-Z_k}{f_{k+1}+f_k} \qquad (9.8)$$

Figure 4 defines k and the order of integration.



Z to ψ and ψ to Z is computed by the program STREAMF (see Appendix B).

Figure 4. Definition of k and order of integration

9.2 INITIALIZATION OF THE SPECIFIC HUMIDITY

Since we are using the ψ-functions as the time-dependent variables, we have to modify the initial humidities in order to make them consistent with the time-integration. Therefore, we put

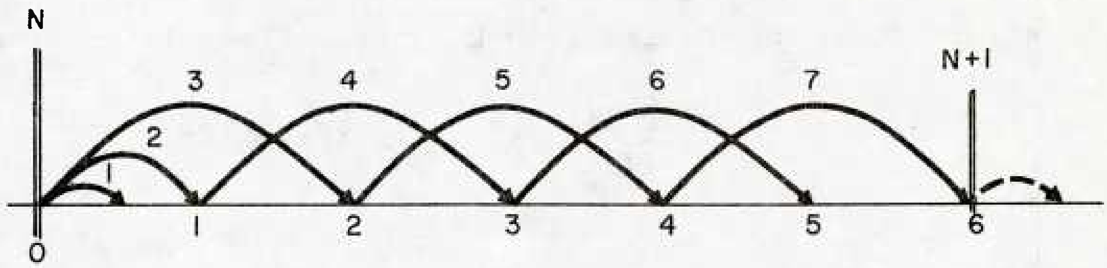
$$q = q_{analyzed} \cdot \frac{q_{sat}\ (\bar{T}_{\psi})}{q_{sat}\ (\bar{T}_{Z})} \tag{9.2.1}$$

$$\bar{T}_{Z} = \frac{2g}{R\ln 2}\left(\frac{Z_{500}-Z_{1000}}{2}\right)=\frac{g}{f_{o}}\ (h_{3}Z_{1}+h_{4}Z_{2}),\text{ and}$$

$$\bar{T}_{\psi} = h_{3}\psi_{1}+h_{4}\psi_{2}.$$

10. TIME-INTEGRATION AND TREATMENT OF LATERAL BOUNDARY VALUES

The forecast is basically computed in 6-hour intervals but this interval can easily be changed. The first time step in each interval is non-centered. Smoothing, elliptization and printing of results (if so desired) are performed at the end of every interval.



$$\frac{\partial\psi}{\partial t} \sim \frac{\Delta\psi}{\varepsilon\Delta t} \qquad \Delta t = 1 \text{ hour in this example} \tag{10.1}$$

ND number of Δt for the interval

kT time step index; $k = 1,2,3 \ldots ND+1$

N index for every interval integration (e.g., 6-hour interval)

N	Forecast length
0	0
1	+6
2	+12
3	+18
.	.
.	.
.	.

Initial height fields $Z°$ are stored on secondary storage during the whole computation. In the case of variable boundary conditions, $Z°$ is followed by Z^N (N=1,2,3...) (forecasts for each interval).

Assuming

$$\frac{\partial \psi}{\partial t} = \frac{g}{f}\frac{\partial Z}{\partial t} \quad \text{or} \quad \psi^N - \psi^o = \frac{g}{f}(Z^N - Z^o) \qquad (10.2)$$

the stream function at time step k can be interpolated from:

$$\psi^k = (\psi^o - \frac{g}{f} Z^o) + \frac{g}{f}[(1-\alpha_k) Z^N + \alpha_k Z^{N+1}] \qquad (10.3)$$

where

$$\alpha_k = \frac{kT-1}{ND}$$

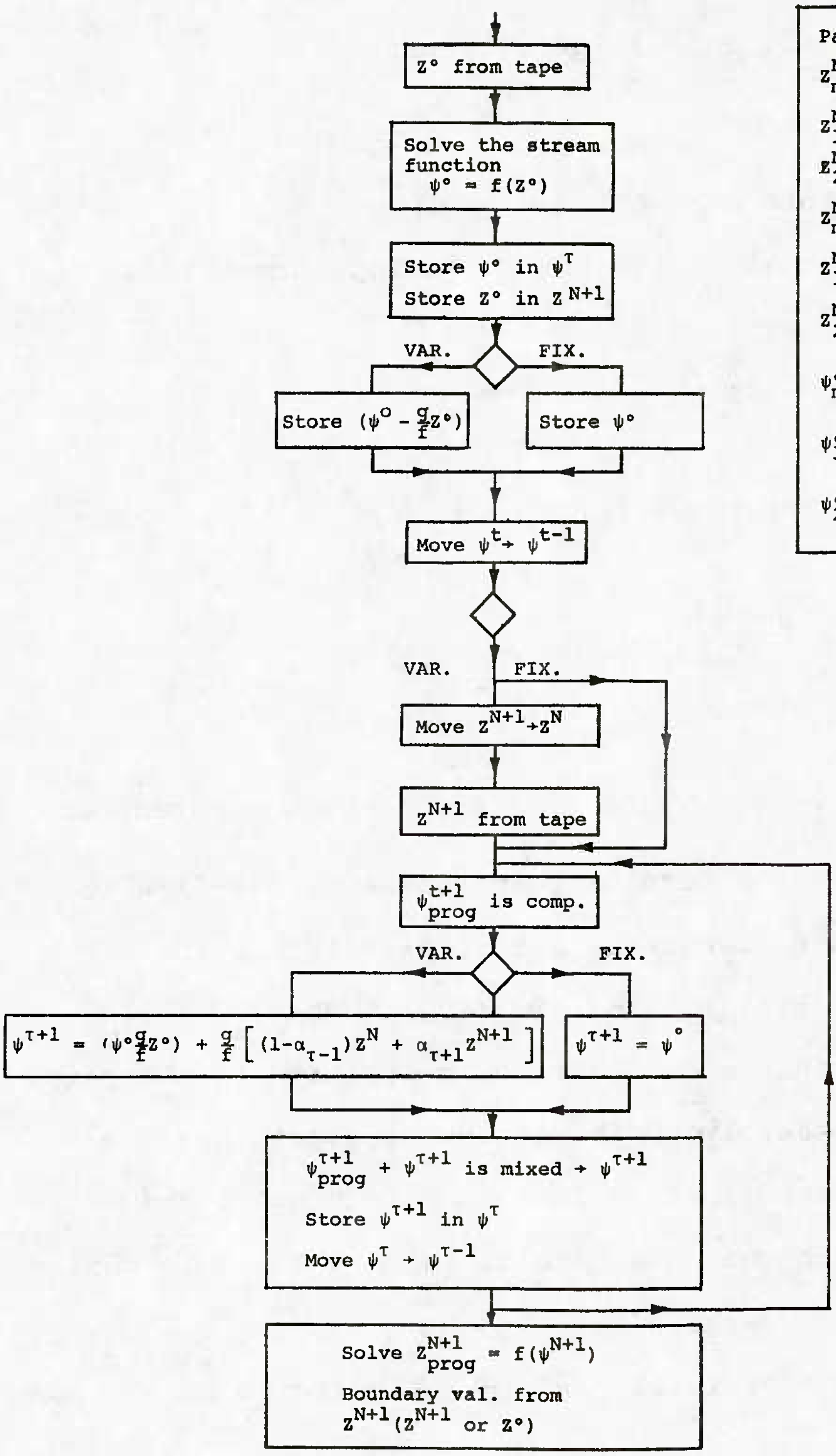
At each time step this interpolated stream function is mixed with the forecasted stream function ψ^k_{prog} in the following way:

Boundary values	$\psi_{mod} = \psi^k_{prog} + w1(\psi^k - \psi^k_{prog})$
1 grid point inside the boundary	$\psi_{mod} = \psi^k_{prog} + w2(\psi^k - \psi^k_{prog})$
2 grid points inside the boundary	$\psi_{mod} = \psi^k_{prog} + w3(\psi^k - \psi^k_{prog})$
3 grid points or more inside the boundary	$\psi_{mod} = \psi^k_{prog}$ (10.4)

w1, w2 and w3 are 3 predetermined constants with $o \leq w1, w2, w3 \leq 1$

The following flow diagram illustrates the treatment of the boundary values:

Flow diagram: Treatment of boundary values.



Prognostic subroutine STEP3P

Parameter	Fieldname
Z_m^N	ZM1
Z_1^N	Z11
Z_2^N	Z21
Z_m^{N+1}	ZM2
Z_1^{N+1}	Z12
Z_2^{N+1}	Z22
$\psi_m^\circ - \frac{g}{f} Z_m^\circ$	STRM
$\psi_1^\circ - \frac{g}{f} Z_1^\circ$	STR1
$\psi_2^\circ - \frac{g}{f} Z_2^\circ$	STR2

II. THE CRITERION OF ELLIPTICITY

An iterative procedure is used to modify a stream-function field so that the ellipticity criterion

$$\nabla^2\psi + \frac{f}{2} > 0$$

is valid in all points of the field.

Each point is tested with the following formula:

$$\mu\nabla^2\psi + \frac{f}{2} - \varepsilon = \delta$$

where

$$\nabla^2\psi = \psi_{i+1,j} + \psi_{i-1,j} + \psi_{i,j+1} + \psi_{i,j-1} - 4\cdot\psi_{i,j}$$

and

$$\varepsilon = 0.001\cdot f.$$

If $\delta < 0$, $\psi_{i,j}$ is modified by

$\psi_{i,j} = \psi_{i,j} - \frac{\varepsilon-\delta}{2\mu} \cdot k$ where k is a convergence parameter.

This means that the vorticity increases by $2(\varepsilon-\delta)\cdot k$ in the point (i,j) and decreases by $0.5\ (\varepsilon-\delta)\cdot k$ in the four surrounding points (i+1,j), (i-1,j), (i,j+1) and (i,j-1). This procedure guarantees that δ becomes positive in the point (i,j), but not necessarily in the surrounding points. The test must therefore be repeated for all points until the criterion is valid in the whole field. With a suitable choice of k the method is convergent.

k can be found empirically and is estimated to be of the order $k \approx 0.85$.

REFERENCES

Bengtsson, L., 1969: Dynamical Weather Prediction, Part B (VI 6). Swedish Meteorological and Hydrological Institute.

Bengtsson, L., and Moen, L., 1971: An operational system for numerical weather prediction. WMO No. 283 Satellite and Computer Applications to Synoptic Meteorology.

Bushby, F.M. and Timpson, M.S., 1967: A 10-level atmospheric model and frontal rain. Quart. J. Roy. Meteor. Soc., 93.

Egger, T., 1972: Incorporation of steep mountains into numerical forecasting models. Tellus, Vol. 24, No. 4.

Gambo, K., 1963: The rôle of sensible and latent heats in the baroclinic atmosphere. J. Meteor. Soc. Japan, 41.

Gerrity, J.P. and McPherson, R.D., 1969: Development of a limited area fine-mesh prediction model. Mon. Wea. Rev., 97.

Reiser, H., 1969: On short-range forecasting with a baroclinic unfiltered model. Proc. WMO/IVGG Symp. NWP, Tokyo.

Shuman, F.G. and Hovermale, T. B., 1968: Operational six-layer primitive model. J. Appl. Meteor.

APPENDIXES

APPENDIX A

GENERAL PRINCIPLES
IN THE
PROGRAMMING OF THE THREE PARAMETER MODEL

The programming has been performed in a very general way. It is therefore possible:

a. to generate initial fields and run the model in a channel with variable cyclic boundary conditions (this is the case for the actual program).

b. to run the model with constant boundary values on a given area (see above).

c. to run the model with variable boundary values, the boundary values taken from another model.

The model needs 13 fields in the core;

1 field for the coriolis parameter	F,
1 field for $\mu = \frac{m^2}{d^2}$	MY,
1 special field indicator	MARK,
10 fields for the model computations	F1-F10.

In addition to this, 40 fields (28 for the quasi-geostrophic part of the model) are specified on secondary memories (disks, drums or large extended core).

"Household" programs:

PUT1: This program computes FPAR (see common, Subroutine JMIMA)

MARKF: This program computes a special integer field MARK. The program MARK specifies status of the field. Points outside the area = 0, points inside the are <0 and points on the boundary >0. The corner points etc. are specified according to given examples.

MYFF: This program computes f and μ and puts the result in the fields F and MY.

RANWT: Writes fields on secondary storage e.g., disk drums, or extended core storage.

RANRD: Reads fields from secondary storage e.g., disk, drums, or extended core storage.

MAP: Prints pattern on line printer; 0 (zero) points, resolution and map scale are specified.

GENCH: Generates initial state for a channel flow.

BMOVE: Administrates computation of cyclic boundary conditions for a channel flow.

Since the axis of the grid is defined differently from what is used in Fortran, the programming of finite difference operators should be performed in a special way. This is not necessary, but speeds up the computation considerably. As an example, subroutine JACOB reads:

```
      .

      .
      .
      .
      .
      MI = M-1
      DØ 10 I = 2,MI
      J1 = JMIN(I)+1
      J2 = JMAX(I)-1
      K = (J1-1)*M+I
      DØ 10 J = J1,J2
      C(K) = (A(K+1) - A(K-1)*(B(K+M) - B(K-M)).....
   10 K = K + M
      .
      .
      .
```

APPENDIX B

PROGRAM SPECIFICATIONS

The following programs are written for the 3-parameter model.

Level 1: Main program

Level 2: Subroutines called by level 1

Level 3: Subroutines called by level 2

Level 4: Subroutines called by level 3

Level	Program
1	PROG3P
2	PUT1
3	JMIMA
2	COEFF3P
2	MARKF
2	MYFF
2	STREAMF
3	BDRGDR
3	BDRVAL
2	SATUR
2	RANWT (RANRD)
2	ELLIPT
2	GENCH
2	ASMUT
2	MAP3P
3	MAP
2	STEP3P

Level	Program
3	JACOB
3	ABSVOR (RELVOR)
3	MIXF
3	HELM (POIS)
3	HELMSYS
3	BMOVE
3	STEPEXT
4	GRADPR
4	VELPOT

Program PROG3 (Main program for 3-parameter model, see flow diagram B-1 and description.)

Arrays

Name	Dimensions	Description
F1-F10	2	Working fields in core. See flow diagram B-1.
F,MY (real)	2	Fields for Coriolis parameter, f, and $\mu = (\frac{m}{d})^2$. m is the mapscale factor for a polar stereographic projection and d is the grid length. F and MY are generated by the subroutine MYFF.
MARK	2	Marking of each grid point by a special integer (index). See description. MARK is generated by the subroutine MARKF.
UPS, UPM, UP1	1	Zonal wind profiles at the levels p_s, p_m and p_1 for the initial fields in the channel case. The values are introduced by DATA statements.
WX	1	Working array for generation of initial fields in the channel case. Dimension shall be at least equal to the number of grid-points across the channel.

Name	Dimensions	Description
NX,NY	1	Wave numbers for perturbations of the initial fields in the channel case in x and y directions. The values are introduced by DATA statements.
PSIC, PSIS, LAMC, LAMS	1	Amplitudes (PSI) and phase lags (LAM) perturbations of the initial field with cosine and sine profiles in the y direction respectively. For both profiles a sine wave is used in the x-direction. The values are introduced by DATA statements. See further description of GENCH.
ZB,PSIB,FB,SB	1	Boundary strings for storage of boundary values in counterclockwise order, starting with corner point 1 (see FPAR). Used by the subroutine STREAMF.
IT	2	Storage of the number of iterations for the solution of a): the system of two Helmholz equations and b): the Helmholz equation for the mean field. Values for an integration interval are stored in the array by the subroutine STEP3P.

Name	Dimensions	Description
MAPHOUR (1)	1	Integers giving the hours for mapping (2).
NSMUTT (1)	1	Integers giving the hours for smoothing (2).
MELLIPT (1)	1	Integers giving the hours for ellipticity (2).
LETACC (1)	1	Integers giving the hours at which the accumulation of precipitation shall be interrupted (2).

---

(1) The values shall be given in DATA statements and must be multiples of the integration interval (e.g., 6 hours). Unused positions are indicated by -1.

(2) This value can be increased if this is necessary.

Flow Diagram B-1: PROG3P

F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	
$z^o(P_s)$	$z^o(P_m)$	$z^o(P_1)$	q^oNA	P_s	T_s	$\psi^o(P_s)$	$\psi^o(P_m)$	$\psi^o(P_1)$		INITF STREAMF
z_m^o	z_1^o	z_2^o	q							
$\rightarrow$	$\rightarrow$	$\rightarrow$								$\rightarrow$ZM2,Z12,Z22 ELLIPT
						x	x	x		
						ψ_m^o	ψ_1^o	ψ_2^o		var. bound.
ψ_m^o	ψ_1^o	ψ_2^o				$\psi_m^o-\frac{g}{f}z_m^o$	$\psi_1^o-\frac{g}{f}z_1^o$	$\psi_2^o-\frac{g}{f}z_2^o$		
ψ_m^o	ψ_1^o	ψ_2^o				ψ_m^o	ψ_1^o	ψ_2^o		
						$\rightarrow$	$\rightarrow$	$\rightarrow$		$\rightarrow$STRM,STR1,STR2
ψ_m^o	ψ_1^o	ψ_2^o	q^o(~50%)	(100.0)	(273.0)	$\psi^o(P_s)$	$\psi^o(P_m)$	$\psi^o(P_1)$		GENCH
$\rightarrow$	$\rightarrow$	$\rightarrow$								$\rightarrow$STRM,STR1,STR2
$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$				0.0	$\rightarrow$PSIM1,PSI11,PSI21, HUM1,PS,TS, 0$\rightarrow$PREC,WS,DIV1,DIV2
0.0										
	x									0$\rightarrow$DIV1,DIV2 HUM1$\rightarrow$HUM2,PSIM1$\rightarrow$PSIM2 PSI11$\rightarrow$PSI12,PSI12$\rightarrow$PSI22 ZM2$\rightarrow$ZM1,Z12$\rightarrow$Z11 Z22$\rightarrow$Z21
$z^{T+1}(P_s)$	$z^{T+1}(P_m)$	$z^{T+1}(P_1)$								
z_m^{T+1}	z_1^{T+1}	z_2^{T+1}								ZINPUT
$\rightarrow$	$\rightarrow$	$\rightarrow$								$\rightarrow$ZM2,Z12,Z22

STEP3P

SMOOTHING

ELLIPT

0$\rightarrow$PREC

Common areas

/FPAR/

Name	Type	Description
IC(8),JC(8)	integer	i- and j-coordinates for the corner points of the area.
XPOL,YPOL	real	i- and j-coordinates for the north pole. Specified relative to the origin of the grid.
R	real	radius of the earth.
RE	real	Distance from the pole to the equator on the map in grid-length units.
DS	real	grid-length (in meters).
JMIN(100), JMAX(100)	integer	minimum and maximum j-coordinate for each i-column in the field.
M,N	integer	Dimension of the field arrays.
KIND	integer	Channel indicator. Channel = 1, No channel = 0. Value shall be given by DATA statement.

/FORM1/

IC1(8),JC1(8)	integer	Same as IC(8),JC(8) for actual field. Values introduced by DATA statement.

Name	Type	Description
JMIN1(100), JMAX1(100)	integer	Same as JMIN(100), JMAX(100)
XP,YP,D1	real	Same as XPOL, YPOL, DS. Values shall be introduced by DATA statements. D1 is given in km.
/KANAL/		
FIM	real	Used for channel computations. Latitude for the middle of the channel. Used for computation of B-plane. To be given by DATA statement.
/DRM/*		
MN	integer	=M*N. To be computed in main program
NDIM	integer	Core memory space available for field arrays. The arrays F1-F10, F,MY,MARK must be included in this area. Additional space is used for storage of fields in COMMON area//.
NFLD	integer	Number of field arrays in core
SL	real	Working area for core memory fields. The dimension must be at least SL(NDIM)

Arrays stored at extended core, disks or drums. 1 in the end of the name indicate timestep τ, 2 in the end of the name timestep τ-1.

---

*Not necessary to specify if only extended core storage is used.

PSIM1,PSI11,PSI21 PSIM2,PSI12,PSI22	real	Storage arrays for stream functions. See flow diagram B-1.
HUM1,HUM2,DIV1 DIV2,WS,HEAT	real	Storage arrays for humidity, divergence, vertical velocity at the lower boundary, and heat.
J789,J12,J56,J3		Storage arrays for Jacobians.
PS,TS,PREC		Storage arrays for surface standard pressure, sea surface temperature, and accumulated precipitation.
STRM,STR1,STR2,ZM1 Z11,Z21,ZM2,Z12,Z22	real	Storage arrays. See Chapter 10 (lateral boundary mixing).
H11,H21,HM1		Fields for advection of vorticity by the divergent wind, thickness field (1,2) and mean field (M).
H12,H22,HM2		$\omega\frac{\partial\zeta}{\partial t}+\zeta\nabla\cdot\mathbb{V}$ for thickness fields (1,2) and mean field (M).
H13,H23,HM3		Twisting term for thickness fields (1,2) and for mean field, M.
V1,V2,VM		Velocity potential for thickness fields (1,2) and for mean field (M).
ID(200)*	integer	Catalog array for direct memory access.

---

*Not necessary to specify if only extended core storage is used.

/COEFF/

A1,A2........,T5	real	Constants used in the model. Computed by the subroutine COEFF3P from given values on stability and pressure levels.

/COEFF2/

T6,TF.......,T38	real	Constants used for the computation of the non-geostrophic terms. Computed by COEFF3P from given values on stability and pressure levels.

/RUNPAR/

DELT	real	Timestep in sec. computed in the main program.
NTSTEP	integer	Number of timesteps for an integration interval (6 hours). To be defined in a DATA statement.
ALFASYS, ALFAM ALFAZ,ALFAPSI, RESSYS,RESM, RESZ,RESPSI	real	Overrelaxation coefficients (ALFA) and maximum residual in the solution for the system of the two Helmholz equations (SYS), Helmholz equation for the mean field equation (M), solution of Z from ψ (Z) and solution of ψ from Z (PSI). To be defined by a DATA statement.

Q,FOCEAN,FCONT	real	The Helmholz term for the mean field equation, friction coefficients over ocean and land. To be defined by a DATA statement.
WGT1,WGT2,WGT3	real	Weights for mixing near the boundary of boundary fields with forecasted fields. To be defined by a DATA statement.
ADIFF	real	Diffusion coefficient for humidity; it is defined by a DATA statement

Parameters only specified in Data statements.

STAB1	real	Static stability of layer 1
STAB2	real	Static stability of layer 2
PNIVS	real	Pressure level P_o
PNIVM	real	Pressure level P_m
PNIV1	real	Pressure level P_1
IVAR	integer	Indicates if variable lateral boundaries are used; IVAR=0, constant boundary values; IVAR=1, variable boundary values.
KIND	integer	Indicates if channel is used (cyclic boundary conditions); KIND=1, channel; KIND=0, no channel (polar stereographic projection).

Subroutine STEP3P(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,MY,F,MARK,
IT,IVAR,M,N)

The subroutine performs a computation for a time interval (6 hours) by the 3P-model including humidity and precipitation prediction. (See Flow Diagram B-2.)

Subroutine Parameters	Description
F1-F10	Working fields.
MY	μ-parameter field.
F	Coriolis parameter field.
MARK	Field indicator field.
IT	Array to store number of iterations for every timestep.
IVAR	Indicator for constant or variable boundary conditions; IVAR = 0 constant boundary; IVAR = 1 variable boundary.
M,N	Field vector.

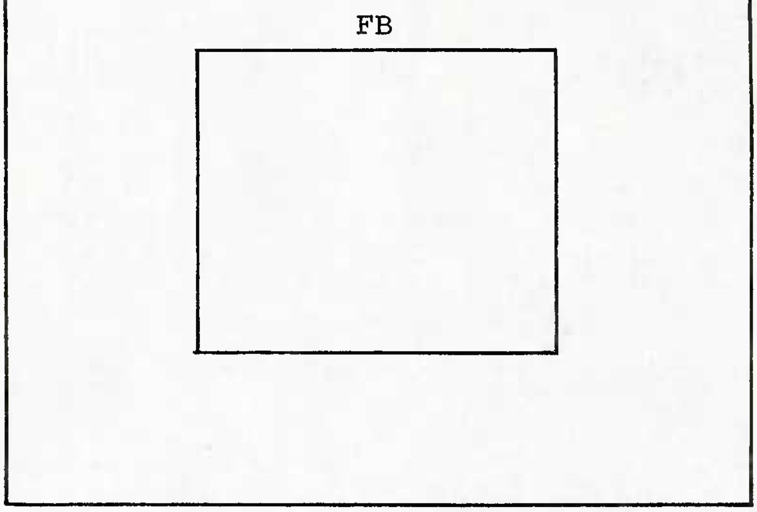
Parameters specified only in DATA statements

RGAS	R = 287
EE	E = 0.622
HL	$L = 2.5 \cdot 10^6$
CP	Cp = 1004
TO	To = 273
EO	Eo = 0.611
DEL1	δ_1
DEL2	δ_2
TOL	Tolerance

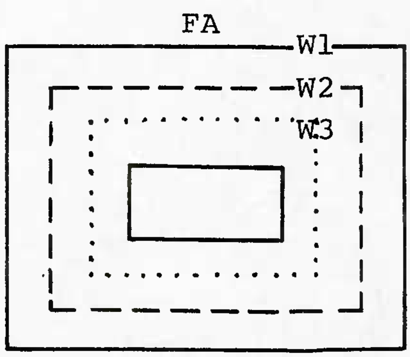
For explanation see chapter 8.4.

Subroutine MARK

The subroutine mixes field FA (corresponding to limited area) with FB (corresponding to a field taken from a large area).



large comp. area



limited comp. area

W_1, W_2, W_3	Weight factors
MN	Field vectors

MARK-field

The MARK-field generated by the subroutine MARKF can be described by the following examples.

1. Channel field.

0	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	0

j ↑ → i

2. Ordinary octagonal field.

0	0	0	0	0	11	10	10	10	10	10	10	10	9	0	0	0	0	0	0
0	0	0	0	12	-4	-8	-8	-8	-8	-8	-8	-8	-3	8	0	0	0	0	0
0	0	0	12	-4	-10	-10	-10	-10	-10	-10	-10	-10	-10	-3	8	0	0	0	0
0	0	12	-4	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	8	0	0	0
0	12	-4	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	8	0	0
13	-4	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	8	0
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	7
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
15	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
0	16	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-2	5
0	0	16	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-2	4	0
0	0	0	16	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-2	4	0	0
0	0	0	0	16	-5	-10	-10	-10	-10	-10	-10	-10	-10	-10	-2	4	0	0	0
0	0	0	0	0	16	-5	-6	-6	-6	-6	-6	-6	-6	-2	4	0	0	0	0
0	0	0	0	0	0	1	2	2	2	2	2	2	2	3	0	0	0	0	0

j ↑ → i

3. Fields with 2 rectangular points.

20	10	10	10	10	10	10	10	10	10	10	10	10	9	0	0	0	0	0	0
14	-9	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-3	8	0	0	0	0	0
14	-9	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-3	8	0	0	0	0
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	8	0	0	0
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	8	0	0
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	8	0
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-3	7
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
15	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
0	16	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
0	0	16	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
0	0	0	16	-5	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
0	0	0	0	16	-5	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-7	6
0	0	0	0	0	16	-5	-6	-6	-6	-6	-6	-6	-6	-6	-6	-6	-6	-7	6
0	0	0	0	0	0	1	2	2	2	2	2	2	2	2	2	2	2	2	18

j

i

0	0	0	0	0	11	10	10	10	10	10	10	10	10	10	10	10	10	10	19
0	0	0	0	12	-4	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-7	6
0	0	0	12	-4	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-7	6
0	0	12	-4	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
0	12	-4	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
13	-4	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-7	6
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-2	5
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-2	4	0
14	-9	-10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-10	-2	4	0	0
14	-9	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-2	4	0	0	0
14	-9	-6	-6	-6	-6	-6	-6	-6	-6	-6	-6	-6	-6	-2	4	0	0	0	0
17	2	2	2	2	2	2	2	2	2	2	2	2	2	3	0	0	0	0	0

j

i

Subroutine RANWT (A,ALFA)

This subroutine writes field A (in core) to field ALFA (in secondary storage).

Subroutine RANRD (A,ALFA)

This subroutine reads field ALFA (in secondary storage) into field A (in core).

Subroutine MIXF(FA,FB,MARK,W1,W2,W3,M,N)

This subroutine mixes field FA (corresponding to limited area) with FB (corresponding to a field taken from a large area).

Subroutine HELMSYS (Z1,Z2,FORC1,FORC2,F,MY,FMY,A1,A2,B1,B2,
ALFA,RESIDUE,IT,M,N)

This subroutine solves the following system of Helmholz equations by relaxation:

$$\nabla^2(Z1) - A1\frac{f^2}{\mu}(Z1) + A2\frac{f^2}{\mu}(Z2) = FORC1$$

$$\nabla^2(Z2) - B2\frac{f^2}{\mu}(Z2) + B1\frac{f^2}{\mu}(Z1) = FORC2$$

Subroutine Parameter	Description
Z1,Z2	Fields to be solved by relaxation. First guesses in Z1,Z2 before using subroutine.
FORC1,FORC2	Forcing functions.
F	Coriolis parameter field.
MY	μ-parameter field.
FMY	Working field.
A1,A2,A3,A4	Physical parameters (constants).
ALPHA	α-overrelaxation coefficient.
RESIDUE	Residual, R, in the solution of the system.
IT	Number of iterations.
M,N	Field vectors.

Subroutine MAP3P(I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,F,MARK,M,N,IDAY,ITIME,NTIME,PNIVS,PNIVM,PNIV1,ZB1,ZB2,ZB3)

Printing on line-printer of forecast fields in zebra patterns. Heights are computed from stream functions.

Subroutine Parameters	Description
I1-I11	Indicators. I = 0: no printing I = 1: printing. The numbers refer to the following fields: I1: Surface pressure. I2: Height for level p_m. I3: Height for level p_1. I4: Thickness (p_m-p_s). I5: Lower vertical velocity $\overline{\omega_1}$. I6: Precipitable water. I7: Accumulated precipitation. I8: Relative humidity. I9: Stream function for level p_s. I10: Stream function for level p_m. I11: Stream function for level p_1.
F1-F10	Working fields. (See flow diagram B-2)
F	Coriolis parameter field.
MARK	Indicator field (see MARKF).
M,N	Field dimension.
IDAY	Year, month and day as one integer.
ITIME,NTIME	Initial and forecast time.
PNIVS,PNIVM,PNIV1	Pressure levels in the model.
ZB1,ZB2,ZB3	Working arrays for boundary strings.

Flow Diagram B-2: MAP3P

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	Variables to and from secondary storage
								ψ_m	ψ_1	ψ_2	PSIM1, PSI11, PSI21
								x	x	x	
	$\psi(P_s)$	$\psi(P_m)$	$\psi(P_1)$								
MAP	.	.	.								
				ω_s	D_1	D_2	q	Σr			WS, DIV1, DIV2, HUM1, PREC
				x	x	x					
				$\overline{\omega_1}$							
							x	x			
						R.H.					
MAP				.		.	.	.			
	x	x	x					Z^T_m	Z^T_1	Z^T_2	ZM2, Z12, Z22
STREAMF					$Z(P_s)$	$Z(P_m)$	$Z(P_1)$				
					x	x					
			P	$2Z_1$							
MAP			.	.		.	.				

Subroutine STREAMF (Z,PSI,R,F,MARK,ZB,PSIB,FB,SB,GAMMA,IND,
IT,M,N)

The subroutine computes the linearized balance equation.

Subroutine Parameter	Description
Z	Z field.
PSI	PSI field.
R	Forcing function for the Poisson equation in the solution of $\nabla^2\psi = F(Z)$ or $\nabla^2 Z = Z(\psi)$.
F	Coriolis parameter.
MARK	MARK-field.
ZB	String of boundary values for Z_k.
PSIB	String of boundary values for ψ_k.
FB	String of boundary values for f_k.
SB	String of boundary values for $(\Delta S)_k$.
GAMMA	γ.
IND	See subroutine comments.
IT	Number of iterations for solving the Poisson equation by relaxation.
M,N	Field vectors.

Subroutines called by STREAMF

BDRGRD(SB)	Computes SB.
BRDVAL(A,AB,M,IND)	Computes boundary values from field A and stores the boundary values in string AB or reverse: IND = 0, AB = A; IND = 1, A = AB.
M	Field parameter
IND	See subroutine comments

POIS is an entry point in HELM for the solution of a Helmholz equation by Liebmann relaxation.

HELM(Z,FORC,Q,ALFA,RESIDUE,IT,M,N)

Subroutine Parameter	Description
Z	Z-field.
FORC	Forcing function.
Q	Helmholz coefficient.
ALFA	Overrelaxation coefficient.
RESIDUE	Residual.
M,N	Field vectors.
IT	Counts the number of iterations required to obtain a solution.

Subroutine SATUR(TM,QSAT)

This subroutine computes QSAT, integrated mixing ratio at saturation as a function of the mean temperature between 500 and 1000 mb. TM values of QSAT are given for each whole degree between -50°C to +20°C in the subroutine. Linear interpolation between whole degrees is employed and the result QSAT is given in TON/m^2/cb.

Subroutine MAP(Q,M,N,DS,QZ,QD,SCALE)

Prints a field in "zebra pattern" on line-printer.

Subroutine Parameters	Description
Q	Field to be printed.
M,N	Field vectors.
DS	Grid distance (meters)
QZ	Isoline corresponding to 000,the line towards 999 on printer (indicated by heavy line below).
QD	Resolution indicator (see below).

```
000000000 \
000000000 /  QD
          \
          /  QD
111111111
111111111

222222222
222222222
```

SCALE	Map scale factor. (unit 10^6 m)

Subroutine JACOB(A,B,C,M,N)

This subroutine computes a Jacobian operator (of the form)

$$C = J(A,B) = (A_{i+1}-A_{i-1})_j (B_{j+1}-B_{j-1})_i$$
$$- (A_{j+1}-A_{j-1})_i (B_{i+1}-B_{i-1})_j \ .$$

Subroutine parameters	Description
A,B,C	Fields according to formula.
M,N	Field vectors .

Subroutine ABSVOR(PSI,VOR,F,MY,MARK,M,N)

This subroutine computes the relative or the absolute vorticity.

a.	$\eta = \mu\nabla^2\psi + f$	ABSVOR
b.	$\zeta = \mu\nabla^2\psi$	RELVOR (via entry point)

Subroutine Parameters	Description
PSI	ψ field.
VOR	η or ζ field.
F	Coriolis parameter field.
MY	μ parameter field.
MARK	Field indicator array.
M,N	Field vectors.

Subroutine GENCH(PSI,M,N,PSIO,U,NU,NWAVE,NX,NY,PSIC,LAMC,
PSIS,LAMS,WX)

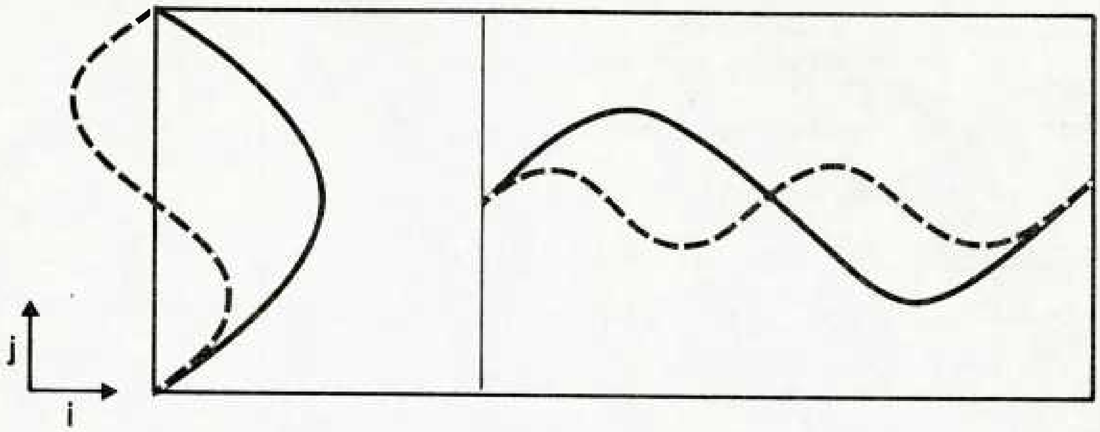
Subroutine Parameters	Description
PSI	Result field.
M,N	Field vectors.
PSIO	Constant value (ψ_o).
U	Zonal wind speed.
NU	Resolution of zonal wind speed.
NWAVE	Number of waves.
NX	Wave numbers as a function of channel length in x-direction (nx).
NY	Wave numbers as a function of channel width in y-direction (ny).
PSIC	Amplitudes of the cosine function (ψ_c).
LAMC	Phase differences for the cosine functions (ψ_c) in whole degrees.
PSIS	Amplitudes of the sine functions (ψ_s) in whole degrees.
LAMS	Phase differences for the sine functions (λ_s) in whole degrees.
WX	Adjustment of the wave near the rigid boundaries: Boundary, WX = 0; +1 row from the boundary, WX = 0.33; +2 row from the boundary, WX = 0.64; +3 and more, WX = 1.

Fields are generated according to the formula:

$$\psi = \psi_o - Uy + \sum_{\nu=1}^{N} [(\psi_c)_\nu \sin\{(nx)_\nu (x) - \frac{\pi}{180}(\lambda_c)_\nu\}\cos(ny)_\nu$$

$$+ (\psi_s)_\nu \sin\{(nx)_\nu (x) - \frac{\pi}{180}(\lambda_s)_\nu\}\sin(ny)_\nu]$$

Example Prediction Area



Nx = 1	1 wave in x-direction over the area (solid line)
Nx = 2	2 waves in x-direction over the area (dashed line)
Nx = N	N waves in x-direction over the area
Ny = 1	1 half wave in y-direction (solid line)
Ny = 2	2 half waves in y-direction (dashed line)
Ny = N	N half waves in y-direction

The zonal wind speed is specified with an arbitrary resolution across the channel. All parameters are given by a DATA statement.

Subroutine STEPEXT(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,MY,F,MARK, M,N,I1,I2,I3,I4)

The subroutine computes the forcing functions

$$\sum_{i=1}^{4} F_{mi}, \quad \sum_{i=1}^{4} F_{1i}, \text{ and } \sum_{i=1}^{4} F_{2i}$$

and stores the result in the fields HM3, H13 and H23 respectively on secondary memories (see Flow Diagram B-3). The program is a subroutine to STEP3P.

Subroutine Parameters	Description
F1-F10	Working field.
MY	μ-parameter field.
F	Coriolis parameter field.
MARK	Field indicator array.
I1,I2,I3,I4	Computational parameters.
	I1 = 0 $\quad \mathbb{V}_\chi \cdot \nabla\eta = 0$
	I1 = 1 $\quad \mathbb{V}_\chi \cdot \nabla\eta \neq 0.$
	I2 = 0 $\quad \zeta \cdot \nabla\mathbb{V} = 0.$
	I2 = 1 $\quad \zeta \cdot \nabla\mathbb{V} \neq 0.$
	I3 = 0 $\quad \omega\frac{\partial\zeta}{\partial p} = 0.$
	I3 = 1 $\quad \omega\frac{\partial\zeta}{\partial p} \neq 0.$
	I4 = 0 $\quad \mathbb{k} \cdot (\frac{\partial\mathbb{V}}{\partial p} x \nabla\omega) = 0.$
	I4 = 1 $\quad \mathbb{k} \cdot (\frac{\partial\mathbb{V}}{\partial p} x \nabla\omega) \neq 0.$
M,N	Field vectors.

Flow Diagram B-3: STEPEXT

F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	To second storage	From second storage	Comments
ψ_m^τ	ψ_1^τ	ψ_2^τ	W_s	Heat	-	-	-	J_4		J4=F8	F5=DIV1 F6=DIV2	
				$D_1^{\tau-1}$	$D_2^{\tau-1}$							
						γ_{10}	γ_{11}	$\nabla\psi_1\nabla\gamma_{10}$	$\nabla\chi_2\nabla\gamma_{11}$			
								F14	F24	H13=F9 H23=F10		Twisting term for thickness fields to second storage
						γ_7	γ_8	$\nabla\psi_1\nabla\psi_7$	$\nabla\psi_2\nabla\psi_8$			
						γ_9	$\nabla\psi_m\nabla\gamma_9$					Twisting term for mean field to second storage.
							F_m^4			HM3=F8		
						ζ_m	ζ_1	ζ_2				
F_{m2}	F_{12}	F_{22}										
$F_{m2}+F_{m3}$	$F_{12}+F_{13}$	$F_{22}+F_{23}$								HM2=F1 H12=F2 H22=F3		$\omega\frac{\partial\zeta}{\partial p}+\zeta\nabla\cdot V$ to secondary storage
$\nabla^2\chi_m$	$\nabla^2\chi_1$	$\nabla^2\chi_2$	$(\chi_m)_g$	$(\chi_1)_g$	$(\chi_2)_g$						F4=VM F5=V1 F6=V2	Compute forcing function for velocity potential in order to get velocity
			χ_m	χ_1	χ_2					VM=F4 V1=F5 V2=F6		
$\eta_m-2\zeta$	$\eta_m+2\zeta_2$											
		$\nabla\chi_m\nabla\zeta_1$							$\nabla\chi_1\nabla\gamma_{13}$			$\gamma_{13}=\eta_m-2\zeta_1$
									F_{11}	H11=F10		$\nabla\chi\nabla\eta$ to secondary storage
		$\nabla\chi_m\nabla\zeta_2$							$\nabla\chi_2\nabla\gamma_{14}$			$\gamma_{14}=\eta_m-2\zeta_2$
									F_{21}	H21=F10		$\nabla\chi\cdot\nabla\eta$ to secondary storage
β_7	β_8	β_9										
						$\nabla\chi_m\nabla\beta_7$	$\nabla\chi_1\nabla\beta_8$	$\nabla\chi_2\nabla\beta_9$				
F_{m1}												
	$F_{m2}+F_{m3}$	F_{m4}	F_{11}	$F_{12}+F_{13}$	F_{14}	F_{21}	$F_{22}+F_{23}$	F_{24}				
ΣF_{mi}			ΣF_{1i}			ΣF_{2i}				HM3=F1 H13=F4 H23=F7		Sum of higher order terms to HM3, H13 and H23
0			0			0						
ψ_m^τ	ψ_1^τ	ψ_2^τ	W_s	Heat				J_4				Restore initial fields

Subroutine VELPOT(KSI,FORC,M,N,RESIDUE,ALFA)

This subroutine computes the velocity potential from a known divergence field

$$\nabla^2\chi = D.$$

In finite-difference form

$$\nabla^2\chi = \frac{D}{\mu}.$$

The solution is performed by Liebmann relaxation with an overrelaxation coefficient ALFA equal approximately to 1.4, but its size depends on the area and mesh width. The residual RESIDUE must also be given ($0.5 \cdot 10^{+6}$ recommended value).

Subroutine Parameters	Description
KSI	2D array for the χ field.
FORC	2D array for the forcing function.
ALFA	Overrelaxation coefficient.
RESIDUE	Residual.
M,N	Field vectors.

Remark: A first guess must be put in χ field before the execution.

Subroutine GRADPR(A,B,C,MARK,M,N)

This subroutine computes a finite difference operation of the form $\nabla A \cdot \nabla B$.

$$(\nabla A \nabla B)_{ij} = (A_{i+1}-A_i)(B_{i+1}-B_i)_j + (A_i-A_{i-1})(B_i-B_{i-1})_j$$

$$+ (A_{j+1}-A_j)(B_{j+1}-B_j)_i + (A_j-A_{j-1})(B_j-B_{j-1})_i$$

Subroutine Parameters	Description
A,B	Fields for operational vector.
C	Result field.
MARK	Field indicator array.
M,N	Field vectors.

APPENDIX C

PROGRAM LISTINGS

Three programs for the three-parameter model in Fortran IV are presented: PROG3P, STEP3P, and STEPEXT. The other subroutines may be obtained from ENVPREDRSCHFAC by request.

PROGRAM PROG3P

```
      PROGRAM PROG3P(OUTPUT,TAPE6=OUTPUT,DATA,INPUT=DATA)                PROG3P.2
* BAROCLINIC BALANCED INTEGRATED 3-PARAMETER MODEL INCLUDING HUMIDITY    PROG3P.3
* AND PRECIPITATION. BOUNDARY VALUES CAN BE VARIABLE, CONSTANT OR OF     PROG3P.4
* CHANNEL TYPE.                                                          PROG3P.5
      DIMENSION F1(57,57),F2(57,57),F3(57,57),F4(57,57),F5(57,57),       PROG3P.6
     1          F6(57,57),F7(57,57),F8(57,57),F9(57,57),F10(57,57),      PROG3P.7
     2          F(57,57),MY(57,57),MARK(57,57)                           PROG3P.8
      DIMENSION UPS(10),UPM(10),UP1(10),WX(15),NX(20),NY(20),PSIC(20),   PROG3P.9
     X             PSIS(20),LAMC(20),LAMS(20)                            PROG3P.10
      DIMENSION ZB(250),PSIB(250),FB(250),SB(250),IT(2,25)               PROG3P.11
      DIMENSION MAPHOUR(10),LETACC(10),NSMUTT(10),NELLIPT(10)            PROG3P.12
      DIMENSION MSMUTT(10)                                               JUN12.2
      DIMENSION MELLIPT(10)                                              JUN12.3
C                                                                        PROG3P.13
C                                                                        PROG3P.14
      COMMON/FPAR/IC(8),JC(8),XPOL,YPOL,R,RE,DS,JMIN(100),JMAX(100),     PROG3P.15
     X             M,N,KIND                                              PROG3P.16
      COMMON/FORM1/IC1(8),JC1(8),JMIN1(100),JMAX1(100),XP1,YP1,D1        PROG3P.17
      COMMON/DRM/MN,NDIM,NFLD                                            PROG3P.18
      COMMON/KANAL/FIM                                                   PROG3P.19
      COMMON F                                                           APR14.2
      COMMON/ECS/ PSIM1,PSI11,PSI21,PSIM2,PSI12,PSI22,HUM1,HUM2,DIV1,    PROG3P.29
     2DIV2,WS,HEAT,J789,J12,J56,J3,PS,TS,PREC,STRM,STR1,STR2,ZM1,Z11,Z21PROG3P.30
     3,ZM2,Z12,Z22,H13,H23,HM3,HM2,H12,H22,H11,H21,J4,VM,V1,V2           PROG3P.31
      COMMON/COEFF/A1,A2,A3,B1,B2,B3,C1,C2,C3,C4,C5,C6,C7,C8,D,DELP,EM,  PROG3P.32
     X            E1,E2,H1,H2,H3,H4,H5,H6,PMEAN,S1,S2,T1,T2,T3,T4,T5,    PROG3P.33
     X            P0,PM,P1                                               PROG3P.34
      COMMON/COEFF2/T6,T7,T8,T9,T10,T11,T12,T13,T14,                     PROG3P.35
     X              K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,  PROG3P.36
     X              K16,K17,K18,K19,K20,K21,K22,K23,K24,K25,K26,K27,K28, PROG3P.37
     X              K29,K30,K31,K32,K33,K34,K35,K36,K37,K38              PROG3P.38
      COMMON/RUNPAR/DELT,NTSTEP,ALFASYS,ALFAM,ALFAZ,ALFAPSI,RESSYS,RESM, PROG3P.39
     X       RESZ,RESPSI,Q,FOCEAN,FCONT,WGT1,WGT2,WGT3,ADIFF             PROG3P.40
      REAL MY                                                            PROG3P.41
      REAL          K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,  PROG3P.42
     X              K16,K17,K18,K19,K20,K21,K22,K23,K24,K25,K26,K27,K28, PROG3P.43
     X              K29,K30,K31,K32,K33,K34,K35,K36,K37,K38              PROG3P.44
      DATA PSIM1,PSI11,PSI21,PSIM2,PSI12,PSI22,HUM1,HUM2,DIV1,           APR14.3
     *DIV2,WS,HEAT,J789,J12,J56,J3,PS,TS,PREC,STRM,STR1,STR2,ZM1,Z11,Z21APR14.4
     *,ZM2,Z12,Z22,H13,H23,HM3,HM2,H12,H22,H11,H21,J4,VM,V1,V2           APR14.5
     */1,3250,6499,9748,12997,16246,19495,22744,25993,29242,32491,35740,APR14.6
     *38989,42238,45487,48736,51985,55234,58483,61732,64981,68230,       APR14.7
     *71479,74728,77977,81226,84475,87724,90973,94222,97471,100720,      APR14.8
     *103969,107218,110467,113716,116965,120214,123463,126712/           APR14.9
      NAMELIST/FPARR/IC,JC,XPOL,YPOL,R,RE,DS,JMIN,JMAX,M,N,KIND          PROG3P.45
      DATA(IC1(I),I=1,8)/1,1,57,57,57,57,1,1/                            PROG3P.46
      DATA(JC1(I),I=1,8)/1,1,1,1,57,57,57,57/                            PROG3P.47
      DATA XP1,YP1,D1 /-21.,29.,95.25/                                   PROG3P.48
      DATA FIM/ 50. /                                                    PROG3P.49
      DATA KIND/0/                                                       PROG3P.50
      DATA ICASE,IDAY,ITIME/1,660922,00/                                 PROG3P.51
      DATA ALFASYS,ALFAM,ALFAZ,ALFAPSI/.85,1.45,1.4,1.8/                 JUN12.4
      DATA RESSYS,RESM,RESZ,RESPSI/.5E5,.5E5,.5, .5E5 /                  PROG3P.53
      DATA Q,FOCEAN,FCONT,WGT1,WGT2,WGT3,ADIFF/   0,0 .,.62,1.,1.,.5,.2, PROG3P.54
     *                                           0,0/                    PROG3P.55
      DATA STAB1,STAB2,PNIVS,PNIVM,PNIV1/.422222,.511111,100.,50.,30./   PROG3P.56
C  NTSTEP IS NUMBER OF TIME STEPS IN 3 HOURS                             APR14.10
      DATA NTSTEP/4/                                                     JUN12.5
      DATA NEND/48/                                                      APR14.12
      DATA IVAR/ 0 /                                                     PROG3P.59
```

```
C Q AND ADIFF HAVE TO BE DEFINED ***                                    PROG3P.60
C MAPHOUR GIVES THE HOURS(MULTIPLE OF 6) FOR MAP-PRINTING.              PROG3P.61
C LETACC GIVES THE HOURS(MULTIPLE OF 6) AT WHICH THE ACCUMULATED PRECIPIPROG3P.62
C SHALL BE PUT =0 AGAIN.                                                PROG3P.63
C NSMUTT GIVES THE HOURS(MULTIPLE OF 6) FOR SMOOTHING.                  PROG3P.64
C NELLIPT GIVES THE HOURS (MULTIPLE OF 6) FOR ELLIPTICITY TEST.         PROG3P.65
      DATA MAPHOUR/0,12,24,36,48,-1,-1,-1,-1,-1/                        JUN12.6
      DATA (LETACC(I),I=1,10)/0,12,24,36,48,-1,-1,-1,-1,-1/             JUN12.7
      DATA(NSMUTT(I),I=1,10)/0,3,6,9,12,15,18,21,24,-1/                 PROG3P.68
      DATA (MSMUTT(I),I=1,10)/27,30,33,36,39,42,45,48,-1,-1/            JUN12.8
      DATA(NELLIPT(I),I=1,10)/0,3,6,9,12,15,18,21,24,-1/                PROG3P.69
      DATA (MELLIPT(I),I=1,10)/27,30,33,36,39,42,45,48,-1,-1/           JUN12.9
      G=9.806                                                           PROG3P.70
      F0=1.03E-4                                                        PROG3P.71
C  TIME STEP IN SECONDS                                                 APR14.14
      DELT=1.*3.6E3/NTSTEP                                              JUN12.10
      LABEL = 10H PUT1                                                  PROG3P.73
      BTIME = SECOND(FAKE)                                              PROG3P.74
      WRITE(6,8000) BTIME                                               PROG3P.75
 8000 FORMAT(1H0, *BTIME=*, F10.4)                                      PROG3P.76
      CALL PUT1                                                         PROG3P.77
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.78
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.79
 8005 FORMAT(1H0, *TIME TO EXECUTE*, A10, F10.4)                        PROG3P.80
      LABEL = 10H COEFF3P                                               PROG3P.81
      BTIME = SECOND(FAKE)                                              PROG3P.82
      WRITE(6,8000) BTIME                                               PROG3P.83
      CALL COEFF3P(STAB1,STAB2,PNIVS,PNIVM,PNIV1)                       PROG3P.84
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.85
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.86
      MN = M*N                                                          PROG3P.87
      NDIM = 20000                                                      PROG3P.88
      LABEL = 10H MARKF                                                 PROG3P.89
      BTIME = SECOND(FAKE)                                              PROG3P.90
      WRITE(6,8000) BTIME                                               PROG3P.91
      CALL MARKF(MARK,M,N)                                              PROG3P.92
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.93
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.94
      LABEL = 10H MYFF                                                  PROG3P.95
      BTIME = SECOND(FAKE)                                              PROG3P.96
      WRITE(6,8000) BTIME                                               PROG3P.97
      CALL MYFF(MY,M,N)                                                 APR14.16
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.99
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.100
      DO 9 I=1,MN                                                       PROG3P.101
    9 F1(I)=0.0                                                         PROG3P.102
      LABEL = 10H RANWT                                                 PROG3P.103
      BTIME = SECOND(FAKE)                                              PROG3P.104
      WRITE(6,8000) BTIME                                               PROG3P.105
      CALL RANWT(VM,F1)                                                 PROG3P.106
      CALL RANWT(V1,F1)                                                 PROG3P.107
      CALL RANWT(V2,F1)                                                 PROG3P.108
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.109
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.110
      NTIME = 0                                                         PROG3P.111
      IF(KIND.NE.0) GO TO 45                                            PROG3P.112
C INITIAL FIELDS. NO CHANNEL.                                           PROG3P.113
C THE SUBROUTINE INITF HAS TO BE WRITTEN ***                            PROG3P.114
C INITIAL Z-FIELDS AT THE LEVELS PS,PM AND P1 TO F1,F2 AND F3. THE HUMIDPROG3P.115
C FIELD Q TO F4. STANDARD SURFACE PRESSURE PS TO F5 AND SEA LEVEL TEMPERPROG3P.116
```

PROG3P (Continued)

```
C TS TO F6, DEFINITION  Q=0.0334892*M(850)+0.0363689*M(700)+0.0290951*M(PROG3P.117
C 0.0145476*M(300) WHERE M(P) ARE MIXING RATIOS IN MTS-UNITS, PS=101.325PROG3P.118
C AT SEA SURFACE.                                                       PROG3P.119
      LABEL = 10H INITF                                                 PROG3P.120
      BTIME = SECOND(FAKE)                                              PROG3P.121
      WRITE(6,8000) BTIME                                               PROG3P.122
      CALL INITF(F1,F2,F3,F4,F5,F6,F7(1,1),F7(1,5),F7(1,9),F7(1,13),    PROG3P.123
     *F7(1,17),F7(1,21),F7(1,25),F8,M,N)                                APR14,17
      SCALE=20.                                                         PROG3P.125
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.126
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.127
      LABEL = 10H MAP 6X                                                PROG3P.128
      BTIME = SECOND(FAKE)                                              PROG3P.129
      WRITE(6,8000) BTIME                                               PROG3P.130
C SOLUTION OF STREAMFUNCTIONS                                           PROG3P.139
      LABEL = 10H STREAMF 1                                             PROG3P.140
      BTIME = SECOND(FAKE)                                              PROG3P.141
      WRITE(6,8000) BTIME                                               PROG3P.142
      LABEL = 10H STREAMF                                               PROG3P.143
      BTIME = SECOND(FAKE)                                              PROG3P.144
      WRITE(6,8000) BTIME                                               PROG3P.145
   15 CALL STREAMF(F1,F7,F10,MARK,ZB,PSIB,FB,SB,GAMMA,0,IT1,M,N)        APR14,18
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.147
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.148
      LABEL = 10H STREAMF 2                                             PROG3P.149
      BTIME = SECOND(FAKE)                                              PROG3P.150
      WRITE(6,8000) BTIME                                               PROG3P.151
      CALL STREAMF(F2,F8,F10,MARK,ZB,PSIB,FB,SB,GAMMA,0,IT2,M,N)        APR14,19
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.153
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.154
      LABEL = 10H STREAMF 3                                             PROG3P.155
      BTIME = SECOND(FAKE)                                              PROG3P.156
      WRITE(6,8000) BTIME                                               PROG3P.157
      CALL STREAMF(F3,F9,F10,MARK,ZB,PSIB,FB,SB,GAMMA,0,IT3,M,N)        APR14,20
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.159
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.160
      DO 20 I=1,MN                                                      PROG3P.161
      F5(I)=F5(I)*0.1                                                   PROG3P.162
      Z=F2(I)                                                           PROG3P.163
      F2(I)=.5*(Z-F1(I))                                                PROG3P.164
      F3(I)=.5*(F3(I)-Z)                                                PROG3P.165
      F1(I)=Z                                                           PROG3P.166
   20 CONTINUE                                                          PROG3P.167
C ADAPTION OF HUMIDITY FIELD                                            PROG3P.168
      DO 21 I=1,MN                                                      PROG3P.169
      TZ = G*(H3*F2(I)+H4*F3(I))/F0                                     PROG3P.170
      TPSI = .5*(H3*(F8(I)-F7(I))+H4*(F9(I)-F8(I)))                     PROG3P.171
      CALL SATUR(TZ,QZ)                                                 PROG3P.172
      CALL SATUR(TPSI,QPSI)                                             PROG3P.173
C CHANGE IN F4 SINCE F4 IS RELATIVE HUMIDITY                            PROG3P.174
      F4(I)=F4(I)*QPSI                                                  PROG3P.175
C     F4(I)=F4(I)*QPSI/QZ                                               PROG3P.176
      Z = F4(I)-.8*QPSI                                                 PROG3P.177
      IF(Z.GT.0.0) F4(I)=.8*QPSI                                        PROG3P.178
   21 CONTINUE                                                          PROG3P.179
C STORE Z-FIELDS FOR BOUNDARY MIXING                                    PROG3P.180
      LABEL = 10H RANWT 3X                                              PROG3P.181
      BTIME = SECOND(FAKE)                                              PROG3P.182
      WRITE(6,8000) BTIME                                               PROG3P.183
      CALL RANWT(ZM2,F1)                                                PROG3P.184
      CALL RANWT(Z12,F2)                                                PROG3P.185
```

```
      CALL RANWT(Z22,F3)                                                PROG3P.186
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.187
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.188
C TEST FOR ELLIPTICITY OF PSI-FIELDS                                    PROG3P.189
      LABEL = 10H ELLIPT 3X                                             PROG3P.190
      BTIME = SECOND(FAKE)                                              PROG3P.191
      WRITE(6,8000) BTIME                                               PROG3P.192
      CALL ELLIPT(F7,MY,M,N)                                            APR14.21
      CALL ELLIPT(F8,MY,M,N)                                            APR14.22
      CALL ELLIPT(F9,MY,M,N)                                            APR14.23
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.196
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.197
C COMPUTATION OF PSIM,PSI1 AND PSI2                                     PROG3P.198
      DO 25 I=1,MN                                                      PROG3P.199
      Z=F8(I)                                                           PROG3P.200
      F8(I)=.5*(Z-F7(I))                                                PROG3P.201
      F9(I)=.5*(F9(I)-Z)                                                PROG3P.202
      F7(I)=Z                                                           PROG3P.203
   25 CONTINUE                                                          PROG3P.204
      IF(IVAR.EQ.0) GO TO 31                                            PROG3P.205
C STORE FIELDS FOR BOUNDARY MIXING                                      PROG3P.206
      DO 30 I=1,MN                                                      PROG3P.207
      Z=F1(I)                                                           PROG3P.208
      F1(I)=F7(I)                                                       PROG3P.209
      F7(I)=F7(I)-G*Z/F0                                                PROG3P.210
      Z=F2(I)                                                           PROG3P.211
      F2(I)=F8(I)                                                       PROG3P.212
      F8(I)=F8(I)-G*Z/F0                                                PROG3P.213
      Z=F3(I)                                                           PROG3P.214
      F3(I)=F9(I)                                                       PROG3P.215
      F9(I)=F9(I)-G*Z/F0                                                PROG3P.216
   30 CONTINUE                                                          PROG3P.217
      GO TO 36                                                          PROG3P.218
   31 DO 35 I=1,MN                                                      PROG3P.219
      F1(I)=F7(I)                                                       PROG3P.220
      F2(I)=F8(I)                                                       PROG3P.221
      F3(I)=F9(I)                                                       PROG3P.222
   35 CONTINUE                                                          PROG3P.223
      LABEL = 10H RANWT 3X                                              PROG3P.224
      BTIME = SECOND(FAKE)                                              PROG3P.225
      WRITE(6,8000) BTIME                                               PROG3P.226
   36 CALL RANWT(STRM,F7)                                               PROG3P.227
      CALL RANWT(STR1,F8)                                               PROG3P.228
      CALL RANWT(STR2,F9)                                               PROG3P.229
      DTIME = SECOND(FAKE) - BTIME                                      PROG3P.230
      WRITE(6,8005) LABEL, DTIME                                        PROG3P.231
C PRINT NUMBER OF ITERATIONS IN THE SOLUTION OF PSI                     PROG3P.232
   41 WRITE(6,202) IT1,IT2,IT3                                          PROG3P.233
  202 FORMAT(///1X,33HNUMBER OF ITERATIONS IN STREAMF ,3(I4))           PROG3P.234
      GO TO 53                                                          PROG3P.235
C GENERATION OF INITIAL FIELDS FOR THE CHANNEL VERSION, PSI-FIELDS ARE  PROG3P.236
C GENERATED IN THE SUBROUTINE GENCH FROM PARAMETERS GIVEN IN DATA-STATEMPROG3P.237
C HUMIDITY-FIELD CORRESPONDS TO 50 PERCENT RELATIVE HUMIDITY, PS IS 100 PROG3P.238
C AND TS 273 DEGREES EVERYWHERE.                                        PROG3P.239
   45 CALL GENCH(F7,M,N,PSIPS,UPS,NU,NWAVE,NX,NY,PSIC,LAMC,PSIS,LAMS,WX)PROG3P.240
      WRITE(6,FPARR)                                                    PROG3P.241
      CALL GENCH(F8,M,N,PSIPM,UPM,NU,NWAVE,NX,NY,PSIC,LAMC,PSIS,LAMS,WX)PROG3P.242
      CALL GENCH(F9,M,N,PSIP1,UP1,NU,NWAVE,NX,NY,PSIC,LAMC,PSIS,LAMS,WX)PROG3P.243
C COMPUTATION OF PSIM,PSI1,PSI2,HUMIDITY,PS AND TS                      PROG3P.244
   46 DO 50 I=1,MN                                                      PROG3P.245
```

PROG3P (Continued)

```
      F1(I)=F8(I)                                                        PROG3P.246
      F2(I)=.5*(F8(I)+F7(I))                                             PROG3P.247
      F3(I)=.5*(F9(I)+F8(I))                                             PROG3P.248
      F5(I)= 100.                                                        PROG3P.249
      F6(I)= 273.                                                        PROG3P.250
      TPSI= H3*F2(I)+H4*F3(I)                                            PROG3P.251
      CALL SATUR(TPSI,QPSI)                                              PROG3P.252
      F4(I)=.5*QPSI                                                      PROG3P.253
   50 CONTINUE                                                           PROG3P.254
C STORE PSI-FIELDS IN STRM,STR1 AND STR2                                 PROG3P.255
      CALL RANWT(STRM,F1)                                                PROG3P.256
      CALL RANWT(STR1,F2)                                                PROG3P.257
      CALL RANWT(STR2,F3)                                                PROG3P.258
C SMOOTHING OF INITIAL FIELDS                                            PROG3P.259
   53 IF(NSMUTT(1).NE.0) GO TO 55                                        PROG3P.260
      LABEL = 10H ASMUT 6X                                               PROG3P.261
      BTIME = SECOND(FAKE)                                               PROG3P.262
      WRITE(6,8000) BTIME                                                PROG3P.263
      CALL ASMUT(F1,F10,M,N, .5)                                         PROG3P.264
      CALL ASMUT(F1,F10,M,N,-.5)                                         PROG3P.265
      CALL ASMUT(F2,F10,M,N, .5)                                         PROG3P.266
      CALL ASMUT(F2,F10,M,N,-.5)                                         PROG3P.267
      CALL ASMUT(F3,F10,M,N, .5)                                         PROG3P.268
      CALL ASMUT(F3,F10,M,N,-.5)                                         PROG3P.269
      DTIME = SECOND(FAKE) - BTIME                                       PROG3P.270
      WRITE(6,8005) LABEL, DTIME                                         PROG3P.271
C LOADING OF INITIAL FIELDS, ZERO TO ACCUMULATED PRECIPITATION           PROG3P.272
   55 DO 56 I=1,MN                                                       PROG3P.273
      F10(I)=0.0                                                         PROG3P.274
   56 CONTINUE                                                           PROG3P.275
      LABEL = 10H RANWT 10X                                              PROG3P.276
      BTIME = SECOND(FAKE)                                               PROG3P.277
      WRITE(6,8000) BTIME                                                PROG3P.278
      CALL RANWT(PSIM1,F1 )                                              PROG3P.279
      CALL RANWT(PSI11,F2 )                                              PROG3P.280
      CALL RANWT(PSI21,F3 )                                              PROG3P.281
      CALL RANWT(HUM1, F4 )                                              PROG3P.282
      CALL RANWT(PS   ,F5 )                                              PROG3P.283
      CALL RANWT(TS   ,F6 )                                              PROG3P.284
      CALL RANWT(PREC ,F10)                                              PROG3P.285
      CALL RANWT(WS   ,F10)                                              PROG3P.286
      CALL RANWT(DIV1 ,F10)                                              PROG3P.287
      CALL RANWT(DIV2 ,F10)                                              PROG3P.288
      DTIME = SECOND(FAKE) - BTIME                                       PROG3P.289
      WRITE(6,8005) LABEL, DTIME                                         PROG3P.290
      IF(MAPHOUR(1).NE.0) GO TO 60                                       PROG3P.291
C MAPPRINTING OF FIRST TIMESTEP                                          PROG3P.292
      IF(KIND.NE.0) GO TO 58                                             PROG3P.293
      LABEL = 10H MAP3P                                                  PROG3P.294
      BTIME = SECOND(FAKE)                                               PROG3P.295
      WRITE(6,8000) BTIME                                                PROG3P.296
      CALL MAP3P(1,1,1,1,0,2,0,2,0,0,0,F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,   APR14,24
     X         MARK,M,N,IDAY,ITIME,NTIME,PNIVS,PNIVM,PNIV1,ZB,PSIB,FB)   PROG3P.298
      DTIME = SECOND(FAKE) - BTIME                                       PROG3P.299
      WRITE(6,8005) LABEL, DTIME                                         PROG3P.300
      GO TO 60                                                           PROG3P.301
   58 CALL MAP3P(0,0,0,0,0,0,0,0,1,1,1,F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,   APR14,25
     X         MARK,M,N,IDAY,ITIME,NTIME,PNIVS,PNIVM,PNIV1,ZB,ZB,ZB)     PROG3P.303
      LABEL = 10H FORECASTS                                              PROG3P.304
      BTIME = SECOND(FAKE)                                               PROG3P.305
```

```
      WRITE(6,8000) BTIME                                              PROG3P.306
C THREE HOURS FORECAST STARTS HERE                                     PROG3P.307
   60 DO 61 I=1,MN                                                     PROG3P.308
      F1(I)= 0.0                                                       PROG3P.309
   61 CONTINUE                                                         PROG3P.310
      CALL RANWT(DIV1,F1)                                              PROG3P.311
      CALL RANWT(DIV2,F1)                                              PROG3P.312
      CALL RANRD(HUM1,F2)                                              PROG3P.313
      CALL RANWT(HUM2,F2)                                              PROG3P.314
      CALL RANRD(PSIM1,F2)                                             PROG3P.315
      CALL RANWT(PSIM2,F2)                                             PROG3P.316
      CALL RANRD(PSI11,F2)                                             PROG3P.317
      CALL RANWT(PSI12,F2)                                             PROG3P.318
      CALL RANRD(PSI21,F2)                                             PROG3P.319
      CALL RANWT(PSI22,F2)                                             PROG3P.320
      IF(IVAR.EQ.0) GO TO 70                                           PROG3P.321
      IF(KIND.NE.0) GO TO 70                                           PROG3P.322
C INPUT OF BOUNDARY FIELD EACH SIX HOUR                                PROG3P.323
C THE SUBROUTINE ZINPUT HAS TO BE WRITTEN ***                          PROG3P.324
      CALL RANRD(ZM2,F2)                                               PROG3P.325
      CALL RANWT(ZM1,F2)                                               PROG3P.326
      CALL RANRD(Z12,F2)                                               PROG3P.327
      CALL RANWT(Z11,F2)                                               PROG3P.328
      CALL RANRD(Z22,F2)                                               PROG3P.329
      CALL RANWT(Z21,F2)                                               PROG3P.330
C     CALL ZINPUT() ZPS,ZPM,ZP1 TO FIELDS F1,F2 AND F3,                PROG3P.331
      DO 65 I=1,MN                                                     PROG3P.332
      Z=F2(I)                                                          PROG3P.333
      F2(I)=.5*(Z-F1(I))                                               PROG3P.334
      F3(I)=.5*(F3(I)-Z)                                               PROG3P.335
      F1(I)=Z                                                          PROG3P.336
   65 CONTINUE                                                         PROG3P.337
      CALL RANWT(ZM2,F1)                                               PROG3P.338
      CALL RANWT(Z12,F2)                                               PROG3P.339
      CALL RANWT(Z22,F3)                                               PROG3P.340
C GENERAL TIMESTEP THREE HOURS AHEAD                                   PROG3P.341
   70 CALL STEP3P(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,MY,MARK,IT,IVAR,M,N)  APR14,26
      NCOUNT =NTIME                                                    PROG3P.343
      NTIME=NTIME+1                                                    JUN12,11
      I1=NTSTEP+1                                                      PROG3P.345
C PRINT NUMBER OF ITERATIONS                                           PROG3P.346
      WRITE(6,200) NCOUNT,NTIME                                        PROG3P.347
      DO 71 I=1,I1                                                     PROG3P.348
      WRITE(6,201) IT(1,I),IT(2,I)                                     PROG3P.349
   71 CONTINUE                                                         PROG3P.350
C SMOOTHING AND ELLIPTICITY TEST                                       PROG3P.351
      I1=0                                                             PROG3P.352
      I2=0                                                             PROG3P.353
      DO 80 I=1,10                                                     PROG3P.354
      IF(ABS(FLOAT(NSMUTT(I))-NTIME).LT.0.1) I1=1                      JUN12,12
      IF(ABS(FLOAT(MSMUTT(I))-NTIME).LT.0.1) I1=1                      JUN12,13
      IF(ABS(FLOAT(MELLIPT(I))-NTIME).LT.0.1) I2=1                     JUN12,14
      IF(ABS(FLOAT(NELLIPT(I))-NTIME).LT.0.1) I2=1                     JUN12,15
   80 CONTINUE                                                         PROG3P.357
      IF(I1.EQ.0.AND.I2.EQ.0) GO TO 95                                 PROG3P.358
      CALL RANRD(PSIM1,F1)                                             PROG3P.359
      CALL RANRD(PSI11,F2)                                             PROG3P.360
      CALL RANRD(PSI21,F3)                                             PROG3P.361
      DO 85 I=1,MN                                                     PROG3P.362
      Z=F1(I)                                                          PROG3P.363
```

PROG3P (Continued)

```
      F1(I)=Z-2*F2(I)                                                   PROG3P.364
      F3(I) = Z+2.*F3(I)                                                PROG3P.365
      F2(I)=Z                                                           PROG3P.366
   85 CONTINUE                                                          PROG3P.367
      IF(I1.EQ.0) GO TO 86                                              PROG3P.368
      CALL ASMUT(F1,F4,M,N, .5)                                         PROG3P.369
      CALL ASMUT(F1,F4,M,N,-.5)                                         PROG3P.370
      CALL ASMUT(F2,F4,M,N, .5)                                         PROG3P.371
      CALL ASMUT(F2,F4,M,N,-.5)                                         PROG3P.372
      CALL ASMUT(F3,F4,M,N, .5)                                         PROG3P.373
      CALL ASMUT(F3,F4,M,N,-.5)                                         PROG3P.374
   86 IF(I2.EQ.0) GO TO 90                                              PROG3P.375
      IF(KIND.NE.0) GO TO 90                                            PROG3P.376
      CALL ELLIPT(F1,MY,M,N)                                            APR14.27
      CALL ELLIPT(F2,MY,M,N)                                            APR14.28
      CALL ELLIPT(F3,MY,M,N)                                            APR14.29
   90 DO 93 I=1,MN                                                      PROG3P.380
      Z=F2(I)                                                           PROG3P.381
      F2(I)=.5*(Z-F1(I))                                                PROG3P.382
      F3(I)=.5*(F3(I)-Z)                                                PROG3P.383
      F1(I)=Z                                                           PROG3P.384
   93 CONTINUE                                                          PROG3P.385
      CALL RANWT(PSIM1,F1)                                              PROG3P.386
      CALL RANWT(PSI11,F2)                                              PROG3P.387
      CALL RANWT(PSI21,F3)                                              PROG3P.388
C MAPPRINTING                                                           PROG3P.389
   95 DO 96 I=1,10                                                      PROG3P.390
      IF(MAPHOUR(I).EQ.NTIME) GO TO 97                                  PROG3P.391
   96 CONTINUE                                                          PROG3P.392
      GO TO 100                                                         PROG3P.393
   97 IF(KIND.NE.0) GO TO 98                                            PROG3P.394
      CALL MAP3P(1,1,1,0,1,1,1,1,0,0,0,F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,  APR14.30
     X          MARK,M,N,IDAY,ITIME,NTIME,PNIVS,PNIVM,PNIV1,ZB,FB,SB)   PROG3P.396
      GO TO 100                                                         PROG3P.397
   98 CALL MAP3P(0,0,0,0,1,0,0,0,1,1,0,F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,  APR14.31
     X          MARK,M,N,IDAY,ITIME,NTIME,PNIVS,PNIVM,PNIV1,ZB,ZB,ZB)   PROG3P.399
C ZERO TO ACCUMULATED PRECIPITATION                                     PROG3P.400
  100 DO 105 I=1,10                                                     PROG3P.401
      IF(LETACC(I).EQ.NTIME) GO TO 106                                  PROG3P.402
  105 CONTINUE                                                          PROG3P.403
      GO TO 115                                                         PROG3P.404
  106 DO 110 I=1,MN                                                     PROG3P.405
      F1(I)=0.0                                                         PROG3P.406
  110 CONTINUE                                                          PROG3P.407
      CALL RANWT(PREC,F1)                                               PROG3P.408
  200 FORMAT(29H1NUMBER OF ITERATIONS BETWEEN,I3,4H AND,I3,6H HOURS)    PROG3P.409
  201 FORMAT(4X,2(I4))                                                  PROG3P.410
  115 IF(NEND.LE.NTIME) STOP                                            PROG3P.411
      GO TO 60                                                          PROG3P.412
      END                                                               PROG3P.413
```

SUBROUTINE STEP3P

```
      SUBROUTINE STEP3P(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,MY,MARK,IT,      APR14.46
     X               IVAR,M,N)                                          STEP3P.3
C SIX HOURS FORECAST BY 3P-MODEL                                         STEP3P.4
      DIMENSION F1(1),F2(1),F3(1),F4(1),F5(1),F6(1),F7(1),F8(1),F9(1),  STEP3P.5
     X          F10(1),MY(1),F(1),MARK(1),IT(2,1)                       STEP3P.6
      COMMON/FPAR/IC(8),JC(8),XPOL,YPOL,R,RE,DS,JMIN(100),JMAX(100),    STEP3P.7
     X          MX,NX,KIND                                              STEP3P.8
      COMMON/ECS/ PSIM1,PSI11,PSI21,PSIM2,PSI12,PSI22,HUM1,HUM2,DIV1,   STEP3P.18
     2DIV2,WS,HEAT,J789,J12,J56,J3,PS,TS,PREC,STRM,STR1,STR2,ZM1,Z11,Z21STEP3P.19
     3,ZM2,Z12,Z22,H13,H23,HM3,HM2,H12,H22,H11,H21,J4,VM,V1,V2          STEP3P.20
      COMMON/COEFF/A1,A2,A3,B1,B2,B3,C1,C2,C3,C4,C5,C6,C7,C8,D,DELP,EM, STEP3P.21
     X             E1,E2,H1,H2,H3,H4,H5,H6,PMEAN,S1,S2,T1,T2,T3,T4,T5,  STEP3P.22
     X             P0,PM,P1                                             STEP3P.23
      COMMON/COEFF2/T6,T7,T8,T9,T10,T11,T12,T13,T14,                    STEP3P.24
     X              K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15, STEP3P.25
     X              K16,K17,K18,K19,K20,K21,K22,K23,K24,K25,K26,K27,K28,STEP3P.26
     X              K29,K30,K31,K32,K33,K34,K35,K36,K37,K38             STEP3P.27
      COMMON/RUNPAR/DELT,NTSTEP,ALFASYS,ALFAM,ALFAZ,ALFAPSI,RESSYS,RESM,STEP3P.28
     X             RESZ,RESPSI,Q,FOCEAN,FCONT,WGT1,WGT2,WGT3,ADIFF      STEP3P.29
      COMMON F                                                          APR14.47
      REAL MY,KEFF                                                      STEP3P.30
      REAL          K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15, STEP3P.31
     X              K16,K17,K18,K19,K20,K21,K22,K23,K24,K25,K26,K27,K28,STEP3P.32
     X              K29,K30,K31,K32,K33,K34,K35,K36,K37,K38             STEP3P.33
      DATA RGAS,EE,HL,CP,T0,E0,DEL1,DEL2,TOL/287.,.622,2.5E6,1004.,     STEP3P.34
     X             273.,.611,0.0,0.0,0.0/                               STEP3P.35
C DEL1,DEL2, AND TOL HAVE TO BE DEFINED         *******                 STEP3P.36
      DEL1=0.0001                                                       STEP3P.37
      DEL2=0.001                                                        STEP3P.38
      TOL=0.0                                                           STEP3P.39
C CONSTANTS FOR COMPUTATION OF LATENT HEAT                              STEP3P.40
      CC1 = 1./T0                                                       STEP3P.41
      CC2 = EE*HL/RGAS                                                  STEP3P.42
      CC3 = EE*HL/CP                                                    STEP3P.43
      CC4 = CC2*CC3                                                     STEP3P.44
      CC5 = DEL1+DEL2                                                   STEP3P.45
C                                                                       STEP3P.46
      IF(KIND.EQ.0) GO TO 5                                             STEP3P.47
      WGT1=1.0                                                          STEP3P.48
      WGT2=.67                                                          JUN12.16
      WGT3=.33                                                          JUN12.17
    5 MN=M*N                                                            STEP3P.51
      ND=NTSTEP+1                                                       STEP3P.52
      M1=M-1                                                            STEP3P.53
      DO 170 KT=1,ND                                                    STEP3P.54
      EPS = 2.                                                          STEP3P.55
      IF(KT.LT.3) EPS = .5*KT                                           STEP3P.56
C                                                                       STEP3P.57
C*****************************JACOBIAN COMPUTATIONS*********************STEP3P.58
C ALL JACOBIANS ARE COMPUTED AND STORED                                 STEP3P.59
      CALL RANRD(PSIM1,F1)                                              STEP3P.60
      CALL RANRD(PSI11,F2)                                              STEP3P.61
      CALL RANRD(PSI21,F3)                                              STEP3P.62
C                                                                       STEP3P.63
      CALL ABSVOR(F1,F4,MY,MARK,M,N)                                    APR14.48
      CALL RELVOR(F2,F5,MY,MARK,M,N)                                    APR14.49
      CALL RELVOR(F3,F6,MY,MARK,M,N)                                    APR14.50
      KEFF=1.0                                                          STEP3P.67
      CALL RANRD(PS,MY)                                                 STEP3P.68
      CALL OROG(F5,MY,M,N,KEFF)                                         STEP3P.69
C                                                                       STEP3P.70
```

STEP3P (Continued)

```
      DO 10 I=1,MN                                                       STEP3P.71
      F7(I) = C3*F4(I)-C2*F5(I)+C4*F6(I) + C7*F(I)                       STEP3P.72
      F8(I) =-C2*F4(I)+C5*F5(I)                                          STEP3P.73
      F9(I) = C4*F4(I)      +C6*F6(I)*2*C7*F(I)                          STEP3P.74
   10 CONTINUE                                                           STEP3P.75
C                                                                        STEP3P.76
      CALL JACOB(F1,F7,F10,M,N)                                          STEP3P.77
      CALL JACOB(F2,F8,F7,M,N)                                           STEP3P.78
      CALL JACOB(F3,F9,F8,M,N)                                           STEP3P.79
      CALL OROG(F7,MY,M,N,KEFF)                                          STEP3P.80
C                                                                        STEP3P.81
      DO 20 I=1,MN                                                       STEP3P.82
      F10(I) = F10(I)+F8(I)+F7(I)                                        STEP3P.83
      F9 (I) = F4 (I)-2*F5(I)                                            STEP3P.84
   20 CONTINUE                                                           STEP3P.85
C                                                                        STEP3P.86
      CALL RANWT(J789,F10)                                               STEP3P.87
C                                                                        STEP3P.88
      CALL JACOB(F2,F9,F7,M,N)                                           STEP3P.89
      CALL OROG(F7,MY,M,N,KEFF)                                          STEP3P.90
      CALL JACOB(F1,F5,F8,M,N)                                           STEP3P.91
C                                                                        STEP3P.92
      DO 30 I=1,MN                                                       STEP3P.93
      F10(I) = F7(I)+F8(I)                                               STEP3P.94
      F9(I) = F4(I)+2*F6(I)                                              STEP3P.95
   30 CONTINUE                                                           STEP3P.96
C                                                                        STEP3P.97
      CALL RANWT(J12,F10)                                                STEP3P.98
C                                                                        STEP3P.99
      CALL JACOB(F3,F9,F7 ,M,N)                                          STEP3P.100
      CALL JACOB(F1,F6,F8 ,M,N)                                          STEP3P.101
      CALL JACOB(F1,F3,F9 ,M,N)                                          STEP3P.102
      CALL JACOB(F1,F2,F10,M,N)                                          STEP3P.103
      CALL OROG(F10,MY,M,N,KEFF)                                         STEP3P.104
C                                                                        STEP3P.105
      CALL RANWT(J3,F10)                                                 STEP3P.106
C                                                                        STEP3P.107
      DO 40 I=1,MN                                                       STEP3P.108
   40 F10(I)=F7(I)+F8(I)                                                 STEP3P.109
      CALL RANWT(J56,F10)                                                STEP3P.110
      CALL RANRD(PS,F10)                                                 STEP3P.111
      PPM=2./(P0-PM)                                                     STEP3P.112
      DO 44 I=1,MN                                                       STEP3P.113
      F7(I)=F1(I)-F2(I)*PPM*(F10(I)-PM)                                  STEP3P.114
      F8(I)=PPM*F5(I)*(F10(I)-PM)-F4(I)*F(I)                             STEP3P.115
   44 F4(I)=F10(I)                                                       STEP3P.116
C                                                                        STEP3P.117
      CALL JACOB(F7,F4,F5,M,N)                                           STEP3P.118
      CALL MYFF(MY,M,N)                                                  APR14.51
C                                                                        STEP3P.120
C******************************LOWER BOUNDARY CONDITION******************STEP3P.121
C INFLUENCE FROM TOPOGRAPHY AND FRICTIO OVERR LAND OR OCEAN SURFACE      STEP3P.122
C OCEAN SURFACE IS ASSUMED WHERE STANDARD PRESSURE PS IS 101.35 CB OR MOSTEP3P.123
      DO 50 I=1,MN                                                       STEP3P.124
      IF(MARK(I)) 49,50,50                                               STEP3P.125
   49 CF=FOCEAN                                                          STEP3P.126
      PP=F4(I)                                                           STEP3P.127
      IF(PP.LT.101.35) CF=FCONT                                          STEP3P.128
      F4(I) = .25*MY(I)*F5(I)+CF*F8(I)                                   STEP3P.129
      F10(I)=PP                                                          STEP3P.130
   50 CONTINUE                                                           STEP3P.131
```

STEP3P (Continued)

```
C
      CALL RANWT(WS,F4)
      CALL RANRD(TS,F5)
C*****************************SENSIBLE HEAT**********************************
C HEATING FROM OCEAN SURFACE WHEN THE AIR IS COLDER, PQ IS THE TEMP DIFF
C OCEAN SURFACE IS ASSUMED WHERE STANDARD PRESSURE PS IS 101.35 CB OR MO
      DO 59 I=2,M1
      J1=JMIN(I)+1
      J2=JMAX(I)-1
      K=(J1-1)*M+I
      DO 59 J=J1,J2
      PP=F10(K)
      PQ=F5(K)-H2*F2(K)
      IF(PP.LT.101.324) GO TO 57
      IF(PQ.LE.0.0) GO TO 57
      PR = SQRT(.25*MY(K)*((F7(K+1)-F7(K-1))*(F7(K+1)-F7(K-1))+
     X              (F7(K+M)-F7(K-M))*(F7(K+M)-F7(K-M))))
      F5(K) = .5E-2*H1*(.1*PR-1.)*PQ
      GO TO 58
   57 F5(K)=0.0
   58 K=K+M
   59 CONTINUE
C*****************************HUMIDITY FORECAST******************************
C EM,E1,E2 ARE COEFF FOR COMP OF MEAN STREAMFUNCTION
C SS1,SS2,SS3 ARE COEFF FOR COMP OF MEAN DIVERGENCE, D IS ZERO OVER LAND
C THE HUMIDITY IS GIVEN IN TON PER SQUAREMETER AND CENTIBAR
      CALL RANRD(DIV1,F6)
      CALL RANRD(DIV2,F7)
      CALL RANRD(HUM1,F8)
C
      SS1 = E1+EM*C2
      SS2 = E2-EM*C1
      SS3 =-EM*C3
      DO 61 I=1,MN
      PQ = D
      PP = F10(I)
      IF(PP.LT.101.35) PQ=0.0
      F7(I) = F8(I)*(SS1*F6(I)+SS2*F7(I)+F4(I)*(PQ+SS3))
      F6(I) = EM*F1(I)+E1*F2(I)+E2*F3(I)
   61 CONTINUE
      CALL JACOB(F6,F8,F10,M,N)
      DO 62 I=2,M1
      J1=JMIN(I)+1
      J2=JMAX(I)-1
      K =(J1-1)*M+I
      DO 62 J=J1,J2
      F6(K) = F8(K+M)+F8(K-M)+F8(K+1)+F8(K-1)-4*F8(K)
      K=K+M
   62 CONTINUE
C
      DEPS = EPS*DELT
      DO 63 I=1,MN
      IF(MARK(I).GE.0) GO TO 63
      F8(I)=-DEPS*(.25*MY(I)*F10(I)+F7(I)-ADIFF*MY(I)*F6(I))
   63 CONTINUE
C
      CALL RANRD(HUM2,F6)
      DO 65 I=1,MN
      IF(MARK(I)) 64,65,65
   64 F6(I) = F6(I)+F8(I)
   65 CONTINUE
C*****************************PRECIPITATION**********************************
```

STEP3P (Continued)

```
C THE RAIN FOR ONE TIMESTEP IS ACCUMULATED. THE RAIN IS GIVEN IN MM OR K
C PER SQUAREMETER
      DO 70 I=1,MN
      IF(MARK(I)) 66,70,70
   66 TEMP = H3*F2(I)+H4*F3(I)
      CALL SATUR(TEMP,QSAT)
      PQ = F6(I)-.8*QSAT
      IF(PQ) 68,68,67
   67 F6(I) =.8*QSAT
      F7(I) = .5E3*DELP*PQ
      IF(KT.EQ.1) F7(I)=0.0
      IF(KT.EQ.2) F7(I)=2*F7(I)
      GO TO 70
   68 F7(I) =.0
      PQ = F6(I)-.2*QSAT
      IF(PQ) 69,70,70
   69 F6(I) =.2*QSAT
   70 CONTINUE
      CALL RANRD(PREC,F8)
      DO 71 I=1,MN
      F8(I) = F8(I) + F7(I)
   71 CONTINUE
      CALL RANWT(PREC,F8)
      CALL RANRD(HUM1,F8)
      CALL RANWT(HUM1,F6)
      CALL RANWT(HUM2,F8)
C****************************LATENT HEAT*********************************
      CALL RANRD(DIV1,F6)
      CALL RANRD(DIV2,F8)
      DO 180 I=1,MN
      IF(MARK(I)) 179,180,180
  179 F6(I) = T1*F4(I)+T2*F6(I)-T3*F8(I)
  180 CONTINUE
C
      DO 190 I=1,MN
      IF(MARK(I)) 187,190,190
  187 RAIN = F7(I)/EPS/DELT
      IF(RAIN.LT.TOL) GO TO 188
      VERT = F6(I)
      IF(VERT.GT.-DEL1) GO TO 188
      OSTAR = VERT
      IF(VERT.GE.-CC5.AND.VERT.LE.-DEL1) OSTAR = -ABS(VERT*VERT/CC5)
      TEMP = H6*F2(I)
      X = CC2*(CC1-1./TEMP)
      E = E0*EXP(X)
      FSTAR = EE*TEMP*E*(CC3-TEMP)/PMEAN/(PMEAN*TEMP*TEMP + CC4*E)
      HLAT = -H1*HL*OSTAR*FSTAR
      GO TO 189
  188 HLAT = 0.0
  189 F5(I) = F5(I) + HLAT
  190 CONTINUE
      CALL RANWT(HEAT,F5)
C
      I1=0
      I2=0
      I3=0
      I4=0

      CALL STEPEXT(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,MY,MARK,M,N,
     X             I1,I2,I3,I4)
C****************************FORCING FUNCTIONS**************************
```

STEP3P (Continued)

```
      CALL RANRD(J789,F10)                                              STEP3P.255
      DEPS = .25*DELT*EPS                                               STEP3P.256
      DO 73 I=1,MN                                                      STEP3P.257
      IF(MARK(I)) 72,73,73                                              STEP3P.258
   72 PQ = 4*F(I)/MY(I)                                                 STEP3P.259
      F8(I) =-DEPS*(F10(I) - C8*F4(I)*PQ)                               STEP3P.260
      F6(I) = PQ*(A3*F4(I)+A1*F5(I))                                    STEP3P.261
      F7(I)=PQ*(B3*F4(I)+B1*F5(I))                                      STEP3P.262
   73 CONTINUE                                                          STEP3P.263
C                                                                       STEP3P.264
      CALL RANRD(J3,F10)                                                STEP3P.265
      CALL RANRD(J12,F4)                                                STEP3P.266
      CALL RANRD(J56,F5)                                                STEP3P.267
C                                                                       STEP3P.268
      DO 80 I=1,MN                                                      STEP3P.269
      IF(MARK(I)) 79,80,80                                              STEP3P.270
   79 PQ = F(I)*F(I)                                                    STEP3P.271
      F6(I) =-DEPS*(F4(I)+PQ*(A2*F9(I)-A1*F10(I))+F6(I))                STEP3P.272
      F7(I) =-DEPS*(F5(I)+PQ*(B1*F10(I)-B2*F9(I))-F7(I))                STEP3P.273
   80 CONTINUE                                                          STEP3P.274
C***************************SOLUTION OF FORECAST EQ.*******************STEP3P.275
      CALL RANRD(HM3,F4)                                                STEP3P.276
      CALL RANRD(H13,F5)                                                STEP3P.277
      CALL RANRD(H23,F10)                                               STEP3P.278
      DEPS1=DELT*EPS                                                    STEP3P.279
      DO 81 I=1,MN                                                      STEP3P.280
      F8(I)=F8(I)+DEPS1/MY(I)*F4(I)                                     STEP3P.281
      F6(I)=F6(I)+DEPS1/MY(I)*F5(I)                                     STEP3P.282
   81 F7(I)=F7(I)+DEPS1/MY(I)*F10(I)                                    STEP3P.283
      CALL RANRD(PSI12,F5)                                              STEP3P.284
      CALL RANRD(PSI22,F10)                                             STEP3P.285
C                                                                       STEP3P.286
      DO 90 I=1,MN                                                      STEP3P.287
   89 F2(I)=2*(F2(I)-F5(I))                                             STEP3P.288
      F3(I)=2*(F3(I)-F10(I))                                            STEP3P.289
   90 CONTINUE                                                          STEP3P.290
C                                                                       STEP3P.291
      LABEL = 10H HELMSYS                                               APR14.57
      BTIME = SECOND(DUMMY)                                             APR14.58
      WRITE(6,8000) BTIME                                               APR14.59
 8000 FORMAT(1H , *BTIME= *, F10.4)                                     APR14.60
      CALL HELMSYS(F2,F3,F6,F7,MY,F4,A1,A2,B1,B2,ALFASYS,RESSYS,        APR14.61
     X            ITSYS,M,N)                                            STEP3P.293
      DTIME = BTIME - SECOND(DUMMY)                                     APR14.62
      WRITE(6,8005) LABEL, DTIME                                        APR14.63
 8005 FORMAT(1H , *TIME TO EXECUTE *, A10, F10.4)                       APR14.64
      IT(1,KT) = ITSYS                                                  STEP3P.294
      CALL ASMUT(F2,F4,M,N,.5)                                          JUN12.18
      CALL ASMUT(F2,F4,M,N,-.5)                                         JUN12.19
      CALL ASMUT(F3,F4,M,N,.5)                                          JUN12.20
      CALL ASMUT(F3,F4,M,N,-.5)                                         JUN12.21
C                                                                       STEP3P.295
      DO 100 I=1,MN                                                     STEP3P.296
      IF(MARK(I)) 99,100,100                                            STEP3P.297
   99 TFILT=0.4                                                         JUN12.22
      IF(MARK(I).EQ.-10) TFILT=0.7                                      JUN12.23
      IF(MARK(I).EQ.-1) TFILT=1.                                        JUN12.24
      F5(I)=F5(I) + TFILT*F2(I)                                         JUN12.25
      F10(I)=F10(I) + TFILT*F3(I)                                       JUN12.26
      F4(I) = Q/MY(I)                                                   STEP3P.300
  100 CONTINUE                                                          STEP3P.301
C                                                                       STEP3P.302
```

```
      CALL RANRD(PSIM2,F6)
C
      DO 110 I=1,MN
  109 F1(I) = 2*(F1(I)-F6(I)) -C2*F2(I) +C1*F3(I)
  110 CONTINUE
C
      CALL HELM(F1,F8,F4,ALFAM,RESM,ITM,M,N)
      IT(2,KT) = ITM
C
      CALL ASMUT(F1,F4,M,N,.5)
      CALL ASMUT(F1,F4,M,N,-.5)
      DO 120 I=1,MN
      IF(MARK(I))119,120,120
119   TFILT=.4
      IF(MARK(I).EQ.-10) TFILT=0.7
      IF(MARK(I).EQ.-1) TFILT=1.
      F6(I)=F6(I) + TFILT*F1(I) + C2*F2(I)
     $-C1*F3(I))
  120 CONTINUE
C*****************************MIXING WITH BOUNDARY FIELDS****************
      CALL RANRD(STRM,F4)
      IF(IVAR.EQ.0) GO TO 156
      IF(KIND.NE.0) GO TO 156
      CALL RANRD(ZM1,F7)
      CALL RANRD(ZM2,F8)
C
      WF = (KT-1)/ND
      FW = 1.-WF
      G = 9.806
      DO 130 I=1,MN
      IF(MARK(I)) 129,130,130
129   F7(I) = G*(FW*F7(I)+WF*F8(I))/F(I)
  130 CONTINUE
      CALL MIXF(F6,F7,MARK,WGT1,WGT2,WGT3,M,N)
      CALL RANRD(STR1,F4)
C
      CALL RANRD(Z11,F7)
      CALL RANRD(Z12,F8)
      DO 140 I=1,MN
      IF(MARK(I)) 139,140,140
139   F7(I) = G*(FW*F7(I)+WF*F8(I))/F(I)
  140 CONTINUE
      CALL MIXF(F5,F7,MARK,WGT1,WGT2,WGT3,M,N)
      CALL RANRD(STR2,F4)
C
      CALL RANRD(Z21,F7)
      CALL RANRD(Z22,F8)
      DO 150 I=1,MN
      IF(MARK(I)) 149,150,150
149   F7(I) = G*(FW*F7(I)+WF*F8(I))/F(I)
  150 CONTINUE
      CALL MIXF(F10,F7,MARK,WGT1,WGT2,WGT3,M,N)
      GO TO 156
  151 CALL MIXF(F6,F4,MARK,WGT1,WGT2,WGT3,M,N)
      CALL RANRD(STRM,F4)
      CALL MIXF(F5 ,F4,MARK,WGT1,WGT2,WGT3,M,N)
      CALL RANRD(STR1,F4)
      CALL MIXF(F10,F4,MARK,WGT1,WGT2,WGT3,M,N)
      CALL RANRD(STR2,F4)
C*****************************STORE NEW TIMESTEP*************************
```

STEP3P (Continued)

```
  156 IF(KT.LT.3) GO TO 157
      CALL RANRD(PSIM1,F1)
      CALL RANRD(PSI11,F4)
      CALL RANRD(PSI21,F7)
  157 CALL RANWT(PSIM1,F6)
      CALL RANWT(PSI11,F5)
      CALL RANWT(PSI21,F10)
      IF(KT.LT.3) GO TO 158
      CALL RANWT(PSIM2,F1)
      CALL RANWT(PSI12,F4)
      CALL RANWT(PSI22,F7)
C****************************COMPUTATION OF DIVERGENCE******************
  158 CALL RANRD(J3,F10)
      CALL RANRD(HEAT,F5)
      CALL RANRD(WS,F8)
C
      PQ = 1./EPS/DELT
      DO 160 I=1,MN
      IF(MARK(I)) 159,160,160
  159 TERM1 = F(I)*(PQ *F2(I)+.25*MY(I)*F10(I))-F5(I)-S1 *F8(I)
      TERM2 = F(I)*(PQ *F3(I)+.25*MY(I)*F9 (I))       -S2 *F8(I)
      F2(I) =- A1*TERM1 + A2*TERM2
      F3(I) = B1*TERM1 - B2*TERM2
  160 CONTINUE
C
      CALL RANWT(DIV1,F2)
      CALL RANWT(DIV2,F3)
      CALL RANRD(PSIM1,F1)
      CALL RANRD(PSI11,F5)
      CALL RANRD(PSI21,F10)
C
      PRINT 7512,KT
 7512 FORMAT(1X,3HKT=,I3)
  170 CONTINUE
      RETURN
      END
```

SUBROUTINE STEPEXT

```
      SUBROUTINE STEPEXT(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,MY,MARK,M,N,     APR14.67
     1                   I1,I2,I3,I4)                                    STEPEXT.3
C THIS SUBROUTINE COMPUTES THE CONTRIBUTION FROM THE HIGHER ORDER TERMS STEPEXT.4
C IN THE VORTICITY EQUATION,                                            STEPEXT.5
C I1 INDICATES THE VORTICITY ADVECTION BY THE DIVERGENT VIND            STEPEXT.6
C I2 INDICATES THE RELATIVE VORTICITY*DIVERGENCE                        STEPEXT.7
C I3 INDICATES THE VERTICAL ADVECTION OF VORTICITY                      STEPEXT.8
C I4 INDICATES THE TWISTINGTERM                                         STEPEXT.9
C I1=0 NO CONTRIBUTION  I1 DIFFERENT FROM 0 CONTRIBUTION FROM I1        STEPEXT.10
C I2=0 NO CONTRIBUTION  I2 DIFFERENT FROM 0 CONTRIBUTION FROM I2        STEPEXT.11
C I3=0 NO CONTRIBUTION  I3 DIFFERENT FROM 0 CONTRIBUTION FROM I3        STEPEXT.12
C I4=0 NO CONTRIBUTION  I4 DIFFERENT FROM 0 CONTRIBUTION FROM I4        STEPEXT.13
C PSIM IS IN F1,PSI1 IS IN F2,PSI2 IS IN F3,WS IS IN F4                 STEPEXT.14
C STEPEXT NEEDS 10 FIELDS IN THE FAST CORE MEMORY F,MY AND MARK MUST    STEPEXT.15
C ALSO BE IN FAST CORE MEMORY                                           STEPEXT.16
      DIMENSION F1(1),F2(1),F3(1),F4(1),F5(1),F6(1),F7(1),F8(1),F9(1),  STEPEXT.17
     1          F10(1),F(1),MARK(1),MY(1)                               STEPEXT.18
      COMMON F                                                          APR14.68
      COMMON/COEFF/A1,A2,A3,B1,B2,B3,C1,C2,C3,C4,C5,C6,C7,C8,D,DFLP,EM, STEPEXT.19
     1             E1,E2,H1,H2,H3,H4,H5,H6,PMEAN,S1,S2,T1,T2,T3,T4,T5,  STEPEXT.20
     2             P0,PM,P1                                             STEPEXT.21
      COMMON/COEFF2/T6,T7,T8,T9,T10,T11,T12,T13,T14,                    STEPEXT.22
     X             K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,  STEPEXT.23
     X             K16,K17,K18,K19,K20,K21,K22,K23,K24,K25,K26,K27,K28, STEPEXT.24
     X             K29,K30,K31,K32,K33,K34,K35,K36,K37,K38              STEPEXT.25
      COMMON/ECS/ PSIM1,PSI11,PSI21,PSIM2,PSI12,PSI22,HUM1,HUM2,DIV1,   APR14.69
     2DIV2,WS,HEAT,J789,J12,J56,J3,PS,TS,PREC,STRM,STR1,STR2,ZM1,Z11,Z21APR14.70
     3,ZM2,Z12,Z22,H13,H23,HM3,HM2,H12,H22,H11,H21,J4,VM,V1,V2          APR14.71
      REAL MY                                                           STEPEXT.34
      REAL         K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,  STEPEXT.35
     X             K16,K17,K18,K19,K20,K21,K22,K23,K24,K25,K26,K27,K28, STEPEXT.36
     X             K29,K30,K31,K32,K33,K34,K35,K36,K37,K38              STEPEXT.37
      KIND=0                                                            APR14.72
      MN=M*N                                                            STEPEXT.38
      RESIDUE =.5E4                                                     STEPEXT.39
      ALFA=1.4                                                          STEPEXT.40
C JACOBIAN J4 TO SECONDARY STORAGE.DIVERGENCIES TO FAST MEMORY          STEPEXT.41
      CALL RANWT(J4,F9)                                                 STEPEXT.42
      CALL RANRD(DIV1,F5)                                               STEPEXT.43
      CALL RANRD(DIV2,F6)                                               STEPEXT.44
      DO 9 I=1,MN                                                       STEPEXT.45
      IF(MARK(I)) 9,7,7                                                 STEPEXT.46
    7 F4(I)=0.0                                                         STEPEXT.47
      F5(I)=0.0                                                         STEPEXT.48
      F6(I)=0.0                                                         STEPEXT.49
    9 CONTINUE                                                          STEPEXT.50
      IF(KIND.EQ.0) GO TO 8                                             STEPEXT.51
      CALL BMOVE(F4,M,N)                                                STEPEXT.52
      CALL BMOVE(F5,M,N)                                                STEPEXT.53
      CALL BMOVE(F6,M,N)                                                STEPEXT.54
    8 CONTINUE                                                          STEPEXT.55
      IF(I4.EQ.0.AND.I3.EQ.0.AND.I2.EQ.0.AND.I1.EQ.0)  GO TO 170        STEPEXT.56
      IF(I4.EQ.0) GO TO 44                                              STEPEXT.57
C COMPUTE THE TWISTINGTERM,I4                                           STEPEXT.58
      DO 10 I=1,MN                                                      STEPEXT.59
      F7(I)=K31*F5(I)+K32*F6(I)+K33*F4(I)                               STEPEXT.60
   10 F8(I)=K36*F5(I)+K37*F6(I)+K38*F4(I)                               STEPEXT.61
      CALL GRADPR(F2,F7,F9,MARK,M,N)                                    STEPEXT.62
      CALL GRADPR(F3,F8,F10,MARK,M,N)                                   STEPEXT.63
      DO 11 I=1,MN                                                      STEPEXT.64
      F9(I)=0.5*MY(I)*F9(I)                                             STEPEXT.65
```

```
   11 F10(I)=0.5*MY(I)*F10(I)                                            STEPEXT.66
      CALL RANWT(H13,F9)                                                 STEPEXT.67
      CALL RANWT(H23,F10)                                                STEPEXT.68
      DO 20 I=1,MN                                                       STEPEXT.69
      F7(I)=K19*F5(I)+K20*F6(I)+K21*F4(I)                                STEPEXT.70
   20 F8(I)=K22*F5(I)+K23*F6(I)+K24*F4(I)                                STEPEXT.71
      CALL GRADPR(F2,F7,F9,MARK,M,N)                                     STEPEXT.72
      CALL GRADPR(F3,F8,F10,MARK,M,N)                                    STEPEXT.73
      DO 30 I=1,MN                                                       STEPEXT.74
   30 F7(I)=K25*F5(I)+K26*F6(I)+K27*F4(I)                                STEPEXT.75
      CALL GRADPR(F1,F7,F8,MARK,M,N)                                     STEPEXT.76
      DO 40 I=1,MN                                                       STEPEXT.77
   40 F8(I)=(F8(I)+F9(I)+F10(I))*0.5*MY(I)                               STEPEXT.78
      CALL RANWT(HM3,F8)                                                 STEPEXT.79
      GO TO 45                                                           STEPEXT.80
   44 DO 46 I=1,MN                                                       STEPEXT.81
   46 F8(I)=0.0                                                          STEPEXT.82
      CALL RANWT(H13,F8)                                                 STEPEXT.83
      CALL RANWT(H23,F8)                                                 STEPEXT.84
      CALL RANWT(HM3,F8)                                                 STEPEXT.85
   45 IF(I2.EQ.0.AND.I3.EQ.0.AND.I1.EQ.0) GO TO 163                      STEPEXT.86
   50 CALL RELVOR(F1,F7,MY,MARK,M,N)                                    .APR14.73
      CALL RELVOR(F2,F8,MY,MARK,M,N)                                     APR14.74
      CALL RELVOR(F3,F9,MY,MARK,M,N)                                     APR14.75
      IF(I2.EQ.0.AND.I3.EQ.0) GO TO 94                                   STEPEXT.90
      IF(I2.EQ.0) GO TO 65                                               STEPEXT.91
C COMPUTATION OF THE RELATIVE VORTICITY*DIVERGENCE,I2                    STEPEXT.92
      DO 60 I=1,MN                                                       STEPEXT.93
      F1(I)=F5(I)*(K1*F8(I)+K2*F9(I)+K3*F7(I))+F6(I)*(K4*F8(I)+K5*F9(I)+STEPEXT.94
     1       K6*F7(I))+F4(I)*(K7*F8(I)+K8*F9(I)+K9*F7(I))                STEPEXT.95
      F2(I)=F5(I)*(K28*F8(I)+F7(I))+F6(I)*K29*F8(I)+F4(I)*K30*F8(I)      STEPEXT.96
   60 F3(I)=F5(I)*K34*F9(I)+F6(I)*(     K35*F9(I)+F7(I))+F4(I)*K30*F9(I)STEPEXT.97
C COMPUTATION OF THE VERTICAL ADVECTION OF VORTICITY,I3                  STEPEXT.98
      IF(I3.EQ.0) GO TO 91                                               STEPEXT.99
      GO TO 80                                                           STEPEXT.100
   65 DO 70 I=1,MN                                                       STEPEXT.101
      F1(I)=0.0                                                          STEPEXT.102
      F2(I)=0.0                                                          STEPEXT.103
   70 F3(I)=0.0                                                          STEPEXT.104
   80 DO 90 I=1,MN                                                       STEPEXT.105
      F1(I)=F1(I)+F5(I)*(K10*F8(I)+K11*F9(I)+K12*F7(I))+                 STEPEXT.106
     1            F6(I)*(K13*F8(I)+K14*F9(I)+K15*F7(I))+                 STEPEXT.107
     2            F4(I)*(K16*F8(I)+K17*F9(I)+K18*F7(I))                  STEPEXT.108
      F2(I)=F2(I)+F8(I)*(K31*F5(I)+K32*F6(I)+K33*F4(I))                  STEPEXT.109
   90 F3(I)=F3(I)+F9(I)*(K36*F5(I)+K37*F6(I)+K38*F4(I))                  STEPEXT.110
      GO TO 91                                                           STEPEXT.111
   94 DO 92 I=1,MN                                                       STEPEXT.112
      F1(I)=0.0                                                          STEPEXT.113
      F2(I)=0.0                                                          STEPEXT.114
   92 F3(I)=0.0                                                          STEPEXT.115
   91 CALL RANWT(HM2,F1)                                                 STEPEXT.116
      CALL RANWT(H12,F2)                                                 STEPEXT.117
      CALL RANWT(H22,F3)                                                 STEPEXT.118
C COMPUTATION OF THE ADVECTION OF VORTICITY BY THE DIVERGENT WIND -I1    STEPEXT.119
C COMPUTE FORCINGFUNCTION FOR THE VELOCITYPOTENTIAL                      STEPEXT.120
      IF(I1.EQ.0) GO TO 166                                              STEPEXT.121
   95 DO 100 I=1,MN                                                      STEPEXT.122
      F1(I)=(C2*F5(I)-C1*F6(I)-C8*F4(I))/MY(I)                           STEPEXT.123
      F2(I) = F5(I)/MY(I)                                                STEPEXT.124
  100 F3(I) = F6(I)/MY(I)                                                STEPEXT.125
C SOLVE THE POISSONEQUATION BY RELAXATION IN ORDER TO GET VELOCITYPOT.   STEPEXT.126
      CALL RANRD(VM,F4)                                                  STEPEXT.127
```

```
      CALL RANRD(V1,F5)                                               STEPEXT.128
      CALL RANRD(V2,F6)                                               STEPEXT.129
      CALL VELPOT(F4,F1,M,N,RESIDUE,ALFA)                             STEPEXT.130
      CALL VELPOT(F5,F2,M,N,RESIDUE,ALFA)                             STEPEXT.131
      CALL VELPOT(F6,F3,M,N,RESIDUE,ALFA)                             STEPEXT.132
      CALL RANWT(VM,F4)                                               STEPEXT.133
      CALL RANWT(V2,F6)                                               STEPEXT.134
      CALL RANWT(V1,F5)                                               STEPEXT.135
      DO 110 I=1,MN                                                   STEPEXT.136
      F1(I)=F7(I)+F(I)-2.*F8(I)                                       STEPEXT.137
  110 F2(I)=F7(I)+F(I)+2*F9(I)                                        STEPEXT.138
      CALL GRADPR(F4,F8,F3,MARK,M,N)                                  STEPEXT.139
      CALL GRADPR(F5,F1,F10,MARK,M,N)                                 STEPEXT.140
      DO 120 I=1,MN                                                   STEPEXT.141
  120 F10(I)=-0.5*MY(I)*(F3(I)+F10(I))                                STEPEXT.142
      CALL RANWT(H11,F10)                                             STEPEXT.143
      CALL GRADPR(F4,F9,F3,MARK,M,N)                                  STEPEXT.144
      CALL GRADPR(F6,F2,F10,MARK,M,N)                                 STEPEXT.145
      DO 130 I=1,MN                                                   STEPEXT.146
  130 F10(I)=-0.5*MY(I)*(F3(I)+F10(I))                                STEPEXT.147
      CALL RANWT(H21,F10)                                             STEPEXT.148
      DO 140 I=1,MN                                                   STEPEXT.149
      F1(I)=C3*(F7(I)+F(I))-C2*F8(I)+C4*F9(I)+C7*F(I)                 STEPEXT.150
      F2(I)=-C2*(F7(I)+F(I))+C5*F8(I)                                 STEPEXT.151
  140 F3(I)=C4*(F7(I)+F(I))+C6*F9(I)+2.*C7*F(I)                       STEPEXT.152
      CALL GRADPR(F4,F1,F7,MARK,M,N)                                  STEPEXT.153
      CALL GRADPR(F5,F2,F8,MARK,M,N)                                  STEPEXT.154
      CALL GRADPR(F6,F3,F9,MARK,M,N)                                  STEPEXT.155
      DO 150 I=1,MN                                                   STEPEXT.156
  150 F1(I)=-0.5*MY(I)*(F7(I)+F8(I)+F9(I))                            APR14,76
      CALL RANRD(HM2,F2)                                              STEPEXT.158
      CALL RANRD(HM3,F3)                                              STEPEXT.159
      CALL RANRD(H11,F4)                                              STEPEXT.160
      CALL RANRD(H12,F5)                                              STEPEXT.161
      CALL RANRD(H13,F6)                                              STEPEXT.162
      CALL RANRD(H21,F7)                                              STEPEXT.163
      CALL RANRD(H22,F8)                                              STEPEXT.164
      CALL RANRD(H23,F9)                                              STEPEXT.165
      DO 160 I=1,MN                                                   STEPEXT.166
      F1(I)=F1(I)+F2(I)+F3(I)                                         STEPEXT.167
      F4(I)=F4(I)+F5(I)+F6(I)                                         STEPEXT.168
  160 F7(I)=F7(I)+F8(I)+F9(I)                                         STEPEXT.169
      GO TO 190                                                       STEPEXT.170
  163 CALL RANRD(HM3,F1)                                              STEPEXT.171
        CALL RANRD(H13,F4)                                            STEPEXT.172
      CALL RANRD(H23,F7)                                              STEPEXT.173
      GO TO 190                                                       STEPEXT.174
  166 CALL RANRD(HM2,F2)                                              STEPEXT.175
      CALL RANRD(HM3,F3)                                              STEPEXT.176
      CALL RANRD(H12,F5)                                              STEPEXT.177
      CALL RANRD(H13,F6)                                              STEPEXT.178
      CALL RANRD(H22,F8)                                              STEPEXT.179
      CALL RANRD(H23,F9)                                              STEPEXT.180
      DO 167 I=1,MN                                                   STEPEXT.181
      F1(I)=F2(I)+F3(I)                                               STEPEXT.182
      F4(I)=F5(I)+F6(I)                                               STEPEXT.183
  167 F7(I)=F8(I)+F9(I)                                               STEPEXT.184
      GO TO 190                                                       STEPEXT.185
  170 DO 180 I=1,MN                                                   STEPEXT.186
      F1(I)=0.0                                                       STEPEXT.187
      F4(I)=0.0                                                       STEPEXT.188
  180 F7(I)=0.0                                                       STEPEXT.189
```

```
190 CALL RANWT(HM3,F1)
    CALL RANWT(H13,F4)
    CALL RANWT(H23,F7)
    CALL RANRD(PSIM1,F1)
    CALL RANRD(PSI11,F2)
    CALL RANRD(PSI21,F3)
    CALL RANRD(WS,F4)
    CALL RANRD(HEAT,F5)
    CALL RANRD(J4,F9)
200 RETURN
    END
```